User Manual

User Manual

Published 1985 ... 2009
Copyright © 1985 ... 2009 Realsoft

All rights reserved. This documentation contains proprietary confidential information and trade secrets of Realsoft.

THIS SOFTWARE AND RELATED MATERIAL IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. IN NO EVENT SHALL REALSOFT BE LIABLE FOR ANY DAMAGES CAUSED USE OF THIS MATERIAL.

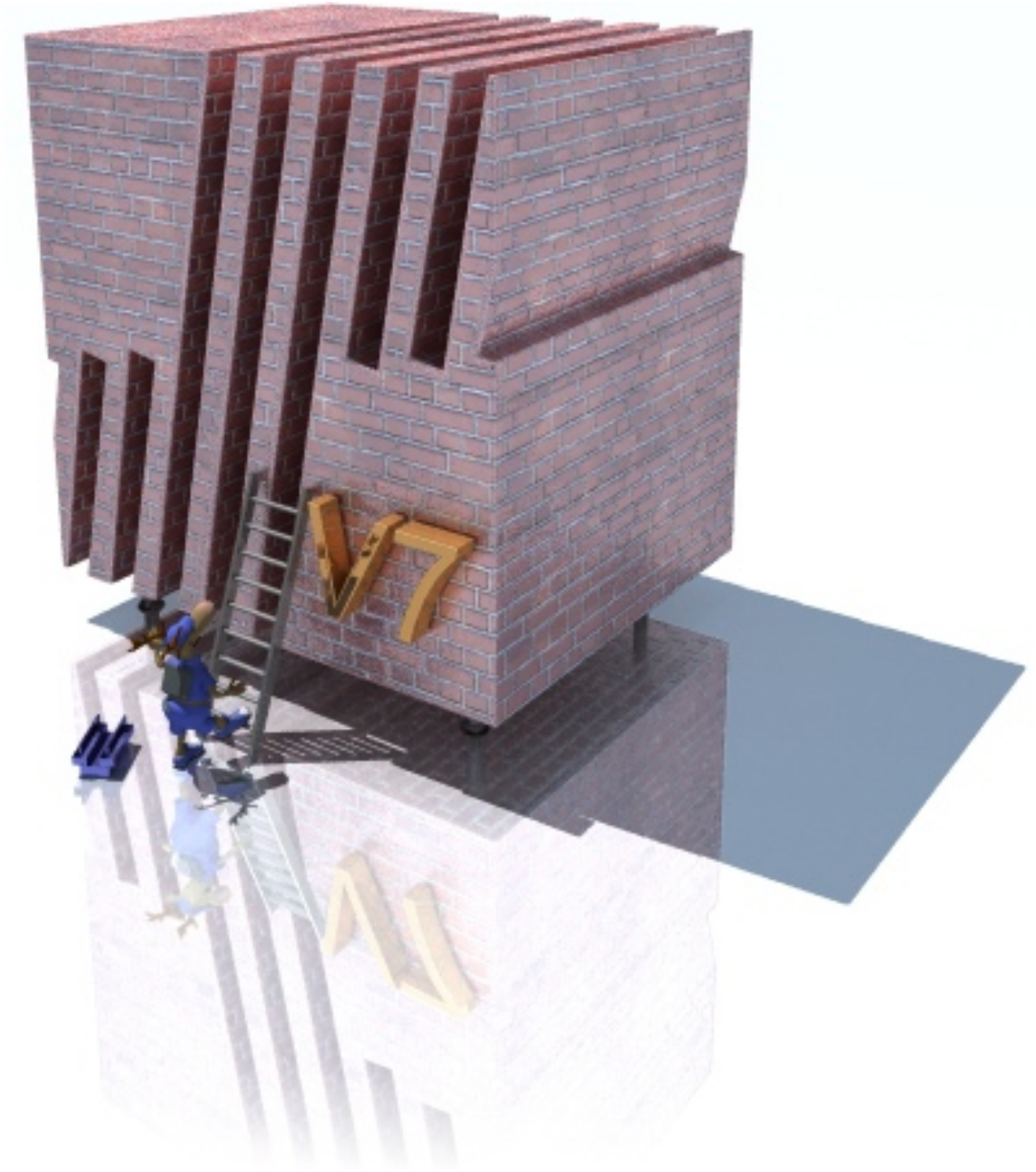
Table of Contents

1. Readme 1
2. Introduction 9
    Getting Started 9
        Installing and Starting the Program 9
        Default User Interface 10
    Building Your First Scene 12
    Modeling Your First Building 23
        Building Walls 23
        Editing the Walls 25
        Windows and Doors 28
        Decorating Rooms 30
        The Roof 31
        Furniture 32
        Landscape 34
    Project Harry 35
        Introduction 35
        Modeling the body 35
        Adding Materials 41
        Programming the Skin with VSL 42
        Local UV Texture 48
        Hair 50
        Skeleton 54
        Animating the Character 58
        Rendering the Animation 62
3. Modeling 65
    Using Curve, Circle and Rectangle Tools 65
    Using Object Handles 71
    Working with Multiple Projects 76
    Snapping to Points Using Dragging 78
    Using the Numeric Interface for Accurate Modeling 81
    Grids 82
    Boolean Operations 83
    Advanced Boolean Operations 88
    Lattice Mapping 89
    Recording and Executing Macros 94
    Particles 97
    Creating Leaves Using 3D Particles 103
    Camera 106
    Fonts 110
    Rotator Object 111
    Tube Object 112
    Instances 113
    Constraints 114
    Field Evaluator 115
    Metaball Object 116
    Link Object 121
4. Subdivision Surfaces 124
    Subdivision Surface Modeling 124
    Using Compass Menus in SDS Modeling 128
    Using Point, Edge and Face Handles 130
    Pointwise Creation of Arbitrary SDS Shapes 132
    Modeling a Bird 134
    Converting Objects to Subdivision Form 136
    Defining Pointwise Properties 140
    Creating Symmetric Objects with the Duplicate Tool 141
    Adding New Points and Faces 144

Rational Subdivision Surfaces 145
Creases 148
Rounding 149
Assigning Materials to Faces 152
Using the Knife tools 155
5. NURBS Curves and Surfaces 157
NURBS Curves 157
NURBS Surfaces 166
Rotational Surfaces 172
Sweep and Bi-Rail Tools 175
Cross Section Surfaces 177
Extruded Surfaces 179
Trimming NURBS Surfaces 180
Welding 184
Fillets 187
The Font Tool 188
Vectorizing Curves from Images 190
Creating Curves from Surfaces 191
The Displace Tool 192
6. Modeling Buildings 194
Creating Parametric Objects 194
Modeling Roofs 206
Multiple Floors 209
Designing Kitchens 212
7. Materials 214
Introduction 214
Blending Materials 218
Texture Mapping 221
Material Classes 224
3D Painting 225
Visual Shading Language 233
Introduction to Visual Shading Language 233
VSL Wizards 238
Building an Interface for a VSL Material 238
Bump Mapping 239
Fog Examples 242
Illumination Examples 244
Miscellaneous Examples 247
Type Casting 251
Parameters 251
Assigning Multiple Materials to SDS Objects 252
Modeling a Pot 256
Modeling the shape 256
Texturing our pot 257
Scope mapping 260
Clip Mapping 262
Previewing Materials 265
Combining 2D and 3D 266
Adding Dirt 273
Adding Dirt 273
Dirty Wall 275
Color Control 277
8. Rendering 281
Introduction 281
Rendering Settings 283
Rendering to a File 283
Distributed Rendering 286
Lighting 288
Volumetric Rendering 294

Motion Blur 299
Field Rendering 301
Scanline Hairs 302
Alpha Channel 305
Advanced Illumination Examples 307
Outline Rendering 314
Caustics 317
Animation with Global Illumination 320
9. Post Effects 329
Post Image Effects 329
Post Particle Effects 332
Blur effect 335
Particle Smoke 336
Compositing 338
Animated Lens Flare 343
Post Shading 344
Technical Outline Drawing Style 348
10. Animations 351
Getting Started with Animation 351
Animation Controls 354
Choreography Basics 356
Choreography Essentials 362
Path Animations 366
Animating Materials 371
Animating Particle Effects 375
Animating Image Effects 375
Rotations 376
Texture Morphing 378
Rendering Animations 380
Skeletons 381
Choreography Parametrization 388
Batch Rendering 393
11. Simulations 394
Simulations Overview 394
Controlling Velocity and Spin 397
The Creator Object 398
Explosions 401
Gravity 403
Collision Detection 406
Electro Magnetism 410
Fluid Dynamics 410
Scripting in Simulations 414
12. User Interface 417
Customizing the Toolbar 417
Customizing the User Interface 418
Binding Macros to Keys 420
Binding Macros to Pull Down Menus 425
Binding Macros to Buttons 425
Binding Macros to Compass Menus 427
Localization 428
13. Scripting 431
Introduction to Scripting Languages 431
Controlling Geometric Objects 431
Binding Scripts to GUI Objects 432
JavaScript 432
Introduction 432
Creating User Interface Objects 433
Managing Geometric Objects 440
Vectors and Matrices 444

Managing Materials 448
Layers 451
Implementing New Classes 452
14. Miscellaneous 457
Hot Keys 457
Productivity Tips 464
Troubleshooting 466
15. Copyright 469
16. Imprint 470
Index 471



Chapter 1. Readme

Welcome to Realsoft 3D Version 7!

If this is the first time you use the software, the tutorials of this User Manual help you to get started. Remember to register at realsoft.com! It qualifies you for technical support, upgrade announcements, special customer discounts and an access to free software updates. After installing the program, you can send us your contact details easily using the pull down menu Help/Realsoft in the Web/Register.

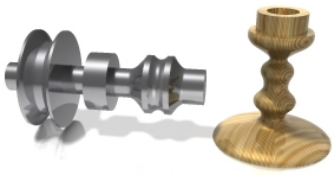
This readme chapter gives a quick overview of what is new in version 7. To keep this introduction easily readable, the new features are described here only briefly. For more information, please consult other sections of the user and reference manuals.

Thank you for choosing Realsoft 3D!



Modeling

A new Lathe tool available. It builds the rotated shape very efficiently using Analytic objects. The tool is easy and straightforward to use, and supports both sharp-edged profile as well as smooth round profiles.



Analytic objects modeled using the lathe tool

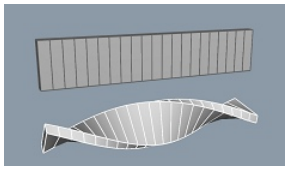
Rounded, sharp and conical tube tools from Real3D V3 plugged in as a single unified tool. The tool provides a quick way to model pipes etc. objects using efficient analytic shapes.



Sharp-edged, rounded and conical analytic tubes combined

The new Twist tool allows easy creation of spiral like forms by twisting a shape around an axis.

Objects can be dropped from the object browse window directly to hierarchy levels in the select window. This is often the quickest way to bring in objects from disk files.



Twist deformation

Metaball object now has a new Expansion attribute. It expands (or shrinks) the included metaballs from their true geometrical size. With the new control, it is easy to set up a collision detection system between metaballs: an expanded size allows objects to melt together, while collision detection prevents excessive penetration.

Metaball level now has an option to scan full sub hierarchy for metaballs. Full hierarchy scan is not always desired, because it can contain parameter objects for animation, such as sample objects for a Creator object.



New metaball options

A new Shift parameter added to plant leaf controls. With this parameter, it is easy to set up useful growth patterns:

- Branch's U Leaf density = 2, Shift = 1 gives regularly alterning leaf pairs
- High U count, Shift = 1 produces a spiral distribution of leaves around the branch



Regularly side switching and spiralwise leaf distributions

Two new Boolean tools, Split and Invert, included.

The 'UV from View' tool, previously available for UV-Set objects, implemented also for the SDS object.

A new option 'Bevel backlid out' available for the NURBS Extrude object.



The new Bevel Backlid Out option blends extruded text smoothly with a base plate

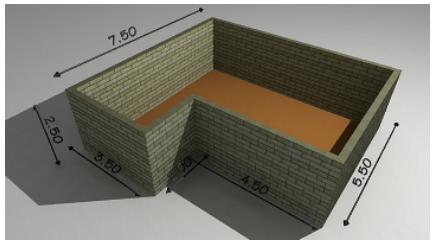


Rendering

Render engine's memory management has been redesigned. The allocated memory is now split into a common, scene specific part, and to a render box (=thread) specific optimization part. This improves memory management dramatically in multi-core machines. A dual core system can typically render 50 % more complex scenes, a quad core 2 times and an eight core system 3-4 times more complex scenes than before. There is no speed penalty, only remarkable speedup when use of VMEM is avoided.

Bottlenecks related to rendering of high poly models with dense render box subdivivison optimized. One million face triset model rendered using 16*12 render boxes renders now almost 10 times faster!

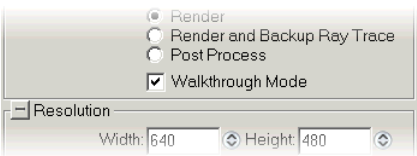
New optimizations speed up ray trace rendering typically 10% - 20% with all kinds of scenes.



Ray tracing -visible measuring lines

Measuring objects and the associated numerical value can be made visible in photorealistic rendering. RT-invisible flag is set by default - clearing it makes the lines visible. Line thickness and rendering method (ray trace/scanline/post process) can be selected from the property window. A new Pseudo 3D option is available, for ensuring a standard size rendering. The option makes the measure objects better readable in all camera angles.

File rendering window now has a new Walkthrough rendering mode. When the walkthrough mode is activated, the program does not redefine the scene in every frame; it is kept static during the whole animation. Only camera changes are considered. This feature can be a real time saver, especially when the scene includes fractal trees or other high-detail geometries. In extreme cases, the walkthrough mode can double the rendering speed.



Fast Walkthrough rendering option

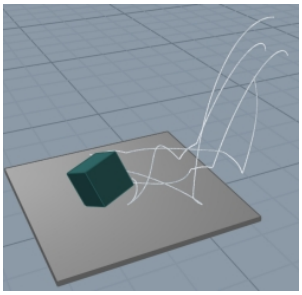
Static option available for mapped shadows. It turns off automatic re-rendering of shadow maps in every render cycle / animation frame.

Terminate ray VSL shader is now evaluated for rays, which are stopped because they fall below recursion threshold. Previously only rays, which hit the recursion depth limit, were treated this way. This allows more detailed shading control and the results are more consistent.



Animation and Modifiers

A general purpose trajectory tracker tool added. It is based on the new modifier approach. The tool can be applied to any animated or simulated object in order to capture its motion as a set of trajectory curves. In point mode trajectories are created for all vertices of the object. In Object mode only one trajectory is created for the object. Three trajectory types supported: continuous, dot and dash (represented as nurbs, 1d or 2d particle objects so that they can later be used as motion curves, exported, or simply used in rendering for artistic impression).

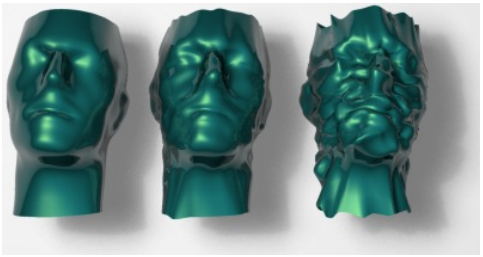


Trajectory curves of a falling cube

Simulation rewinding system improved. The initial state is no longer restored by reading a backup file, but using the construction history approach. For example, modifications applied in the middle of the simulation will not disappear after rewind.

Warp modifier tool. Any geometric object can define a deformation effect. For example, using an analytic sphere, one can deform a planar NURBS mesh to a sphere.

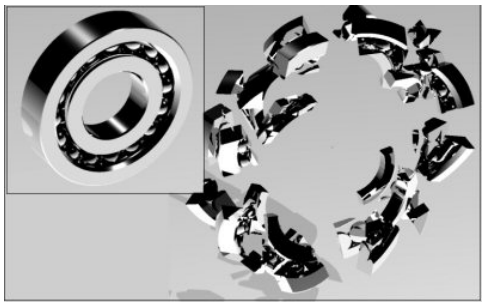
The new Noise modifier deforms targets by recursive fractal 3D noise. Combining random deformations with other animation effects is now really easy.



Noise modifier in action

Morphing object rewritten to use the modifier approach, featuring:

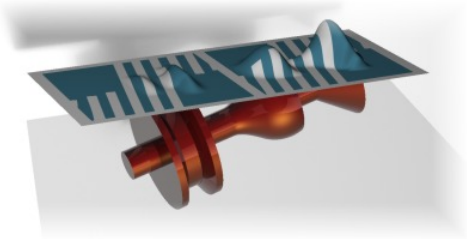
- Any attribute can be morphed
- A single morphing modifier can be bound to any number of target objects
- Morphing effect can be made relative similar to all modifiers. Several morphing modifiers can control a single target, each with its own weight
- Morphing effect can be repeated cyclically, extrapolated, auto closed as a loop or stopped at the extreme key values



Ball bearing exploded volumetrically

Explosions are now implemented using the modifier principle. The new system introduces many new options, such as true volumetric division of exploding volume using boolean operated instances, successive recursive explosions, and division to cubes, spheres or surface elements.

Real3D v3 style shrink wrap implemented as a general purpose collision detection modifier. Three projection types supported: parallel, spherical and cylinder-wise.



Animated shrink wrapping

The new Axis modifier defines its deformation using three factors: tangential (rotation), radial (scale) and axial translation. The level of deformation depends on the distance/projection to specified axis as well as on the selected control curve: linear, hyperbolic, spherical or gaussian. For example, the closer the particle to the axis the higher the rotation.



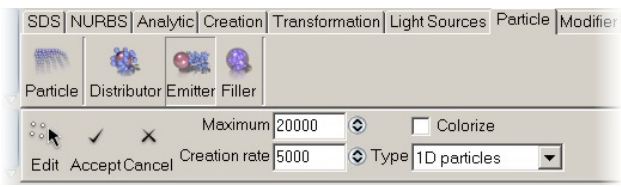
Twist effect animated using the general purpose axis deformer



Particles

A tool for creating particle emitters from geometric objects. Emitter can now generate particles colored by the emitting surface. The volume of the particle flow can be adjusted and turned off easily using the new creation rate attribute.

A tool for filling geometric objects with particles



New particle tools

A tool for distributing particles over surfaces. Particle color can be defined by the color of the corresponding surface point. Creation rate control included.

All particle types now include controls for easy access for particle's age at rendering stage. By default, each particle's age grows relatively from zero to one during its lifetime, regardless of the actual time (in seconds) the particle existed.

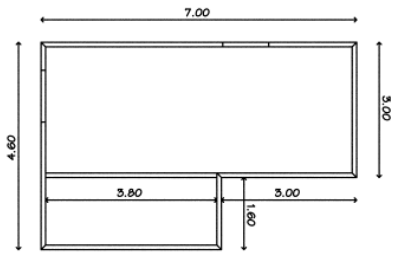


Import / Export

Full support for ILM's OpenEXR image format. The new format supports arbitrary number of channels and layers, and all data types - half, integer and float - of OpenEXR. The format provides a widely supported solution for creating HDR illumination effects.

PostScript (ps) and EmbeddedPostScript (eps) export are now included

View window has a new popu menu: Render/Save 2D Projection. It stores the 2D projected contents of the view into a file using a selected export format, such as dxf or postscript. The file can be edited further in a CAD program or printed.



Top view projection exported as 2D vectors

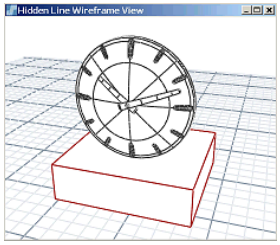


User Interface

Grid object has new controls for dynamic snapping. When user zooms out and the grid becomes too dense, it automatically changes the snapping interval. There is also a density based color mapping option available, similar to the ground plane drawing: small grids are drawn lighter than coarse grids. Basically, now a single grid can work well in all modeling scales and distances.

View window has a new Ruler option. The ruler shows the scale of the model by displaying a line segment and its length using the current distance measuring unit. The ruler is especially useful with a visible grid, because the ruler unit is automatically matched to the current grid. In version 7, the density of a grid may change automatically when zooming, and the ruler helps to keep track of the grid scale.

New View controls introduced:



Hidden line wireframe view

- View window's sync time (interrupt threshold) can be defined accurately. This helps to optimize the responsiveness of the user interface for each computer system. Previously there was just a switch for selecting a short or a long sync time.
- View's backgound color can be changed per view.
- Object selection highlight color can be defined per view

Using these controls, some useful new view's has been configured to the View pull down menu:

- Scrollable high resolution view for saving print resolution black and white wireframe images
- Black and white hidden line wireframe view

View saving to image file now has a new interface, which allows saving to all supported image formats. Previously saving was limited to bmp.



The select window now has a new 'Find Selected' pop-up menu, which autoscrolls the select window to show the currently selected objects.

Select window specific methods, such as 'Rename object' and the new 'Find selected' can be recorded to macros and bound to keys, buttons or compass menus for maximum ergonomy

The select window's Filter field is hidden by default maximizing the space for the actual select tree. Selecting the 'Set Filter' menu shows the field. It then remains visible as long as there is a non-empty filter string defined in the field.

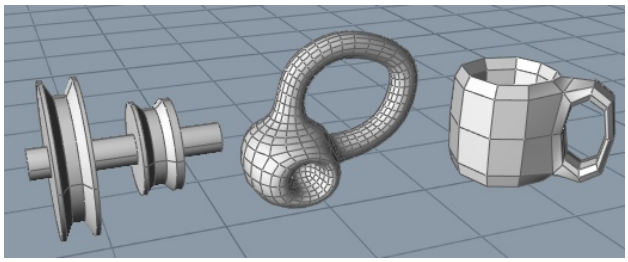
All select window tabs use now the hierarchical tree gadget similar to the geometric objects tab. Objects can be multi selected and hierarchical objects can be plugged in.



Real Time Rendering

OpenGL view's now have 4 drawing modes available:

- Wire
- Shaded
- Shaded+Wire
- Hidden Line Wireframe


Analytic, smooth SDS and polygon objects displayed in Shaded+Wire composite mode

Object drawing quality controls (Wire, Shaded, Texture) and Forced Visibility work now hierarchically. Changing the quality of a hierarchy level updates all children.

Texture quality control has now an explicit UV Image option, which selects quick display of the UV texture (defined in Property window/Col tab/Texture map controls) in its native resolution. Evaluated 3D materials are not shown in this mode. Quality = None option hides all textures, including the UV texture.



Wire profiles + shaded OpenGL

Object wireframes have a new option 'Draw Only Profile'. If set, only wireframe of sharp edges and the object boundary is drawn. This works especially well when visualizing dense SDS/NURBS objects in shaded OpenGL. The reduced wireframe only enhances the main features of the geometry.



Miscellaneous

Materials now have a 'Disable' switch. Using it, effect of a material can be removed and restored back from the whole scene easily.

Two new VSL Wizards for Fade Map and Illumination Map added.

Undo buffer is now chronological: last applied operation is undone regardless of the current select window layer. Operations changing several layers - such as project loading - can be undone with a single undo step.

A new proportional unit type implemented. Unit choices are plain decimal, per cent and per mill points. The proportional scale is a conveninet choice for numeric controls, which display values between 0..1. Per mill scale controls, such as rendering quality tresholds, display useful values, even when the user has chosen only 2 or 3 decimal accuracy

New key event handling featuring the good old "any-key-as-modifier" and the new "exec-by-key-down" ideas. This new implementation works in all platforms (unlike the original v5 implementation which worked only on Windows).

Curve editor now provides 3 extrapolation choices:

- Flat (default setting, used in earlier versions)
- Linear (curve is extended tangetially)
- Repeat (shape is repeated multiple times)



Flat, linear and repeating extrapolations

The feature is used e.g. in GI post system. GI gamma curve now handles all illumination ranges consistently.

The scripting interface now supports new type r3boundingbox. The methods that deal with bounding boxes now work in javascript. A bounding box Defines two vector attributes: min and max.



Chapter 2. Introduction

Getting Started

Installing and Starting the Program

To install the program, browse the contents of the install CD-ROM disk or the folder where you downloaded the setup file. Double click the 'Realsoft 3D-7.0-x64.exe' setup icon and follow the instructions.

Realsoft 3D can be started in the same way as any standard application, by clicking the Window's Start button and selecting the menu: All Programs # Realsoft # Realsoft 3D # Realsoft 3D

After installation, you need to set the install key. Start the program and select the pull down menu Help/About. The opened About window includes an Install Key field. Type your install key into it exactly as written in the sticker of the install CD or in the license registration message, and then click the Set Key button. Note that characters are case sensitive. You can also find the install key after the download link in the Software Updates area of realsoft.com.



Note

The install key consists of your personal license key, and a platform specific extension. It should look as follows:

```
annnnnnnnnnnn.nnnnnnn
```

If you do not set the install key, the program will expire after 2 months. It will then run in demo mode, save functionality disabled.

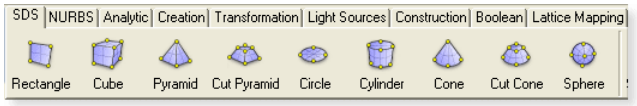
If you are a new Realsoft 3D user, remember to register your license! It qualifies you for technical support, upgrade announcements, special customer discounts and an access to free software updates at

www.realsoft.com. After installing the program, you can register easily from the pull down menu Help/Realsoft in the Web/Register.

Default User Interface

When you start the program, the default user interface is loaded from the startup project. The default interface consists of the following elements:

Toolbar

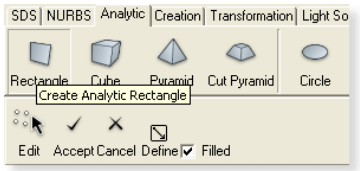


Tabbed Toolbar

The Toolbar allows you to activate such things as the creation and modification tools, as well as many others. The default configuration shows a 'tabbed' toolbar, with tool icons grouped under several tabs like SDS, Creation, Light Sources, and so on. You can hide/show the toolbar by clicking the header area at the top left corner of the toolbar.

Control Bar

The Control Bar is located right below the Toolbar. The control bar is context sensitive. When you select a tool (the Rectangle tool, for instance), the control bar contains the set of modifiers that are relevant to that tool. If you activate the Rectangle tool, valid rectangle-specific options are shown.



Rectangle tool selected. The control bar automatically shows the tool-specific controls

The Control Bar always shows options that are related to the action the user is currently performing. Therefore, its contents change frequently, and adapt to the workflow.

The Control Bar contain more tools than may fit into your screen. However, you can scroll the toolbar with the mouse wheel or by dragging it with the right mouse button while holding down the Ctrl key.

View Control Bar

The View Control Bar is a tall and narrow vertical window, located near the right edge of the main window. It allows you to control the view orientation, and other options specific to View Windows.



For example, the View Control Bar contains buttons for switching between Front/Side/Top views, activating different rendering methods, and rotating, zooming, and panning a view.



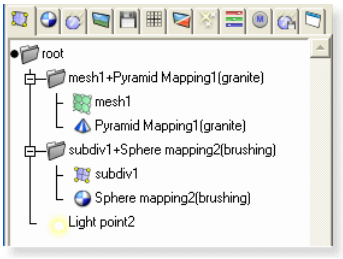
Note

Clicking the Front, Side and Top button while holding the **Shift** down will switch the view into the opposite direction.

The View Control Bar can be scrolled in the same way as the Toolbar if all the controls do not fit into the screen area.

Select Window

The Select Window is located near the left edge of the main window. It allows you to select objects based on their names. When working with large projects, this window is extremely useful for selecting all types of objects, from geometric ones to light sources and material maps. Through its various tabs, the Select window also allows you to manage such things as materials, rendering settings, post effects, and so on.



The Select Window

The Select Window allows you to manage the object hierarchy using a drag&drop interface. For example, you can drag one object inside another 'level' object. A drag box can be used for multi-selecting objects. You can also multi-select objects by holding down the **Shift** key while clicking objects.

If there are more objects than fit into the window, you can scroll the contents of the select window in the same way you can scroll the toolbar: just hold down the **Ctrl** key while dragging with the right mouse button.

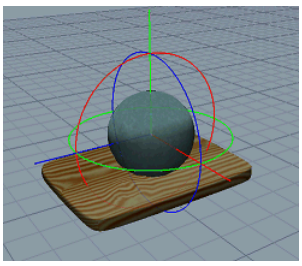
View Window

The View Window is perhaps one of the most often needed windows. It renders geometric 3D objects using various rendering methods, such as wire frame, shaded OpenGL, or ray tracing. You can enter 3D data through the View Window using mouse or some other input device, such as a digital tablet.

To Pan a view window (scroll it): **Ctrl** + drag right mouse.

Zoom a view window: **Shift** + drag right mouse vertically

To Rotate a view window: **Alt** + drag right mouse.



View in Shaded OpenGL mode

Status Window

The Status Window is where the program prints out messages and other feedback.



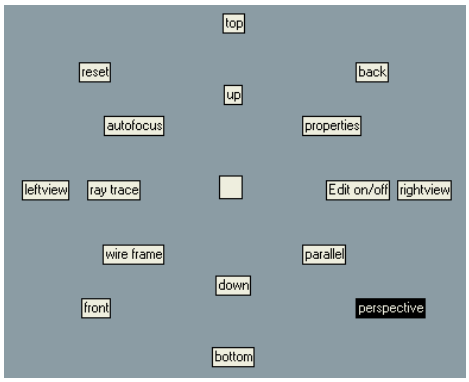
Status window

In addition to these, there are several other windows included with the standard startup project. You can open any of these windows through the View pull down menu.

Compass Menus

View window also support special purpose menus called Compass Menus. They are context sensitive. The contents of the compass are usually set to include those functions that are most likely needed in each situation. For example, if a curve object is selected, the compass menu includes the most frequently needed curve editing functions.

You access a Compass Menu by dragging the right mouse button in the view window. Where you drag, the mouse determines which compass menu is displayed. When you have moved the mouse more than a certain threshold distance, the compass menu is opened.



Drag with the right mouse button to select far south-east compass menu

For example, to enable the perspective view projection, drag while pressing the right mouse button in the View Window. The compass menu is displayed. Keep moving the mouse until the perspective label is highlighted and release the mouse button.

The big idea behind the compass menus is that your muscles can learn to use them. Selecting functions in this manner can become very fast after some practice, because it is not necessary to watch the menu while using it (unlike when using traditional menus, or buttons where you have to accurately position the mouse pointer over the desired item). Your hand remembers the necessary motions to pick an item.

Compass menus also can be bound to keys. For example, the View window compass menu (shown in the previous illustration) is bound to **v** key. Holding down the **v** key while dragging with the right mouse button overrides the default context sensitivity rules, allowing you to access the view specific options whenever you need.

Compass menus are an extremely powerful way to access tools in Realsoft 3D and we strongly suggest you learn to use them.

Building Your First Scene

This first tutorial project introduces you the basic workflow for building and rendering a simple animation in Realsoft 3D. It is intended for first time users including users familiar with previous Realsoft products, because it introduces several features not available in previous program versions.

In this tutorial, you will do the following:

- Model a wooden table with a marbled sphere on it
- Add a light source
- Create a camera
- Animate the sphere to move along the table
- Render the animation to an AVI an MPEG file.

Tutorial level: Beginner

Modeling

To prepare the View Window for modeling the table, make sure OpenGL Shading and Perspective options are set in the View Control Bar. Depending on the screen resolution, you may have to scroll the View Control Bar all the way down. You can either drag with the LMB when the mouse is not over any of the gadgets of the View Controls bar (the cursor changes to an up/down arrow) or hold down the Ctrl while dragging with the right mouse button anywhere on the View Controls bar. Or simply use the Mouse Wheel for scrolling



The first task is to model a simple rectangular table with a simple set of legs. All five of the pieces needed for the table can be constructed using the Analytic Cube tool. Let us start by creating the tabletop first.

We create the tabletop by defining it from the top view. To switch the view to the top view, click the Top button near the top edge of the View Controls Bar.



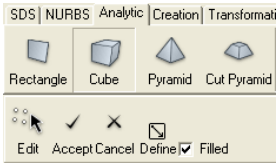


Tip

Shift modidier swaps the direction 180 degrees, so that you can also switch to Back, Right and Bottom views using the same view controls.

Make sure you are viewing the Analytic tab in the Toolbar, then click the Cube icon with the left mouse button to activate the tool.

The Control bar now displays options related to creating cubes. The default settings are fine for the current task.

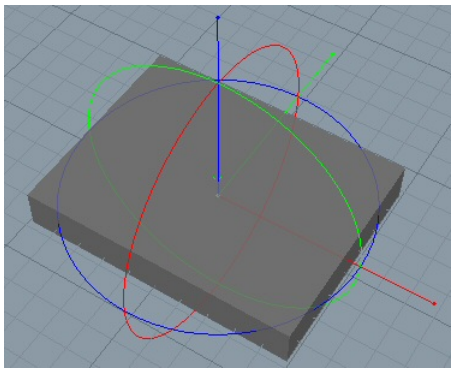


The Cube Tool activated

The Definition type is Corner-Corner, meaning you define the upper left corner and the lower right corner of the rectangle. Filled is checked, so a closed cube with six sides will be constructed.

Click the Left Mouse Button once near the top left corner of the view window to define the upper right corner (do NOT hold down the LMB, just click). Move the mouse down to the right bottom corner of the view. Click with the LMB button again. A cube is created.

Controlling the Orientation of the View Window



Rotate the view by dragging with the right mouse button while holding down the Alt key.

To verify that you really created a cube, you might want to rotate the view window a bit.

You can rotate the view window interactively, using the mouse and a key modifier. To rotate, zoom, or pan the View Window, hold down either the Alt (rotate), Ctrl (pan), or Shift (zoom) key and drag with the right mouse button.

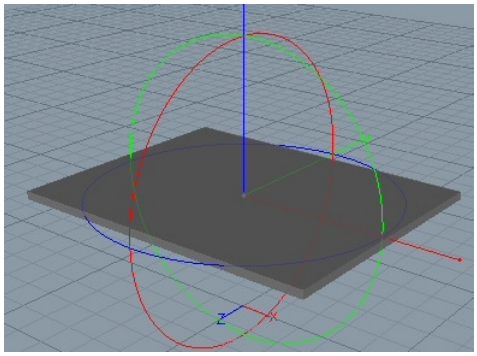
There are several other ways for you to alter the View window or change its orientation. The view window's compass menu is one alternative. Hold down the 'v' key and drag the RMB in the view window until the compass menu appears. The outermost items are view orientation settings. Here, in addition to the Front, Top, and Left-side views, you can also find menus to rotate the current view left, right, up and down 90 degrees.



Compass menu

Transforming

The three circles and axes associated with the created cube are so called transformation handles. These handles are shown for all selected object and you can drag them to move, rotate or scale the object.



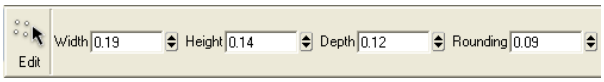
Cube translated and scaled in 'y' direction

The created cube appears to be slightly too thick, so let's scale it in its 'y' direction. To do this, move the mouse over the end point of the vertical 'y' axis handle. Then press the left mouse button key down to start scaling. Move the mouse downwards to see how the object gets scaled in its y direction. Release the mouse button when you are happy with the thickness of the tabletop.

The tabletop also lies now on the ground plane, so let's try moving it up a bit. To do this, move the mouse over the actual 'y' axis handle. Then press the left mouse button down and move the mouse upwards to lift the tabletop. Release the mouse to exit the move tool.

Similary, you would be able to rotate objects using the three cirular rotation handles.

When you select an object, the Control Bar (the context sensitive tool bar) shows you the most commonly needed tools for editing the selected object. In this case, it shows you the controls for editing the size and the edge roundings of the cube.



Cube specific controls

Most real world objects have slightly rounded edges. By default, the program displays measures using meter scale. Therefore, set the Rounding field to a small value, such as 0.005 = 5 mm. The preferred measuring unit can be selected using File/Preferences pull down menu.

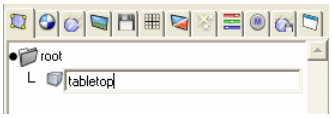
Note that the view window doesn't show you the rounding in any way. The rounding will show up later when we create an image using photo realistic ray tracing.

Renaming

The Select window at the left side of the screen has several tabs on it. Currently, it is displaying the Geometric Objects tab, which shows the hierarchical structure of the object geometry of your project.

In general, selecting objects from the Select window is more accurate and far less error prone than trying to select objects in a view window.

The default name for created objects is typically the name of the object type (cube for instance) followed by a number. Since this is not a good name name for our tabletop, we'll change it.



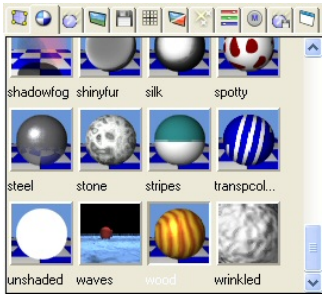
Rename the cube as tabletop

Move the mouse over the cube object in the select window, and then left click the name slowly two times. You can also select Rename from the pop-up menu of the select window. The default name of the level is highlighted and placed in an edit box. Type the word "tabletop" and press the Enter key.

Using Materials

The next step is to assign a 'wood' material with our tabletop object. Make sure the tabletop object is still selected and then click on the Materials tab. The select window now displays the current material library.

Click the Wood material to select it (you may need to scroll the material list first). Next, click the right mouse button over the wood material icon to open a popup menu. This popup menu is different from the menu that was available from the Select window. (The popup menu is context sensitive, and its contents depend on the place where you open it.)



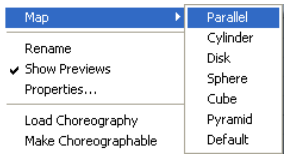
Select the 'wood' material



Note

If you see only a list of material names instead of the preview images, click with the right mouse button and select Show Previews from the popup menu.

There is a Map submenu at the bottom of the popup menu. Move the cursor down the list to highlight the Map option. A submenu opens that displays the available mapping object types.

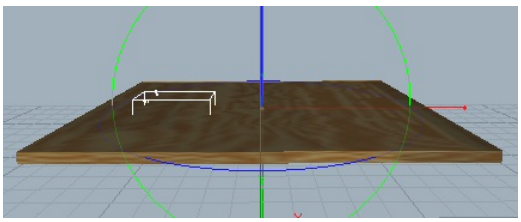


Select Parallel Map Tool

A mapping object wraps a material to a target object according to a specific map geometry, which can be cube-like, disk-like, cylindrical, spherical, etc. Select Parallel mapping from the menu.

Which one is most suitable depends on both the characteristics of the material and the shape of the target object. The Wood material is designed so that it works best with parallel mapping.

Then in the view window, switch to the Side view and define the parallel map geometry with two mouse clicks. Just make the parallel mapping object smaller to achieve suitably dense wood grains.



The white parallel map box defines wood grain projection



Note

The direction, from which you draw the parallel map (the Side view above) defines how wood grains travel trough the tabletob.

The Hierarchy

Click the Geometric Objects tab (first on the left) of the Select window. Notice that the structure has changed; now, instead of a "tabletop" object under the root, you have a new folder called "tabletop + Parallel Mapping(wood)". Open this folder by clicking the little box at the left side of the name.

You can see that the new folder contains the tabletop object and a parallel mapping object. When you drew the parallel map of the Wood material for the tabletop object, the mapping tool created a new hierarchy level and placed the wood mapping and the target cube into that level.



Tabletop consists of a cube and a wood mapping

Realsoft 3D uses the concept of hierarchy extensively to manage the data a user places in the scene. For example, a human body consists of several parts: arms and legs, a head, a torso, etc. In turn, an arm has several subparts like upper arm, forearm, and hand; a hand consists of several more parts, and so on. The object hierarchy is designed to categorize and collect object groups, in the same way that the file system on your computer does.

Similarly, the wooden tabletop in this example consists of a tabletop shape and a wood mapping object. Select the 'parallel mapping(wood)' object. If you want bigger wood grains, just scale the mapping object bigger. If you want to control the direction of the wood grain, rotate the mapping object. After adjusting the mapping, click the WF-Inv button in the MiscTools bar below the select window to hide the wireframe of the parallel map. We do not need to see it any more, so hiding it helps to keep the scene visually cleaner and simpler.



Tip

A mapping object only affects other objects that are in the same hierarchy level!

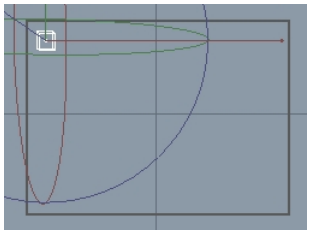
The next step is to create four legs for the table.

Modeling a wooden leg for our table is easy, now that we know how to create a wooden tabletop so we repeat the steps very briefly now.

Switch to the top view. The legs will be created under the tabletop. The shaded tabletop now blocks our view under the table, making further modeling hard. Therefore, turn Shaded OpenGL drawing off using the view control icon in the lowest group of the view control bar.



Turn Shaded OpenGL off



Create a cube representing a leg

Look at the select window; you should see so called Current Level marker (by default, a black dot) at the left side of the 'tabletop+parallel mapping(wood)' name. The Current Level defines where the program places new objects. A leg is not a part of the tabletop, so click the root folder to make it the current level.

Enable the Cube tool. Then enter two points in the view window, somewhere close a tabletop corner, to create a cube representing a leg.

Rename the created cube as 'leg'.

You may find it quite difficult to place the cube properly using perspective view. As intuitive as the perspective view is, it makes accurate modeling with the mouse difficult. Turn off the Perspective option either using View popup/Camera/Perspective menu or from the View control bar's Perpective icon. Then switch to the front view. Use the transformation handles to scale the leg longer and fit it under the tabletop. The parallel view projection makes these steps easy and accurate.

Switch back to the top view. Attach wood the material to the leg exactly the same way as you attached it to the tabletop. This time use the top view when drawing the parallel map, to align the wood grains with the leg. If you used the side view, the wood grain would be perpendicular to the leg, which is quite unnatural.

Now the 'root' object consists of two sub objects: a wooden tabletop and a wooden leg.

Duplicating

Rather than making three more legs from scratch, we can just duplicate the one we already have. To do this, make first sure the 'leg + parallel map(wood)' object is selected, then click the RMB in the Select window, and choose Duplicate from the popup menu.

Then drag one of the translation handles in the view window to move the newly created leg to the opposite edge of the tabletop.

Rather than duplicating the remaining legs one by one, we can multi select both legs, duplicate and translate them in one simple operation.



To multi select the two legs, hold down the Shift key and click each leg object with the mouse. You can click them either in the select window or in the view window.

Legs multiselected



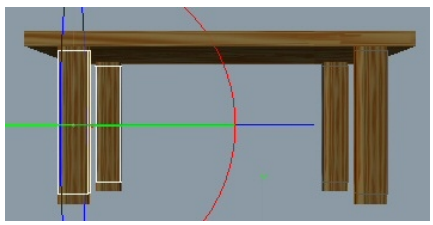
Tip

Realsoft 3D supports the following selection modifiers:

- Shift - multi select
- Ctrl - select range
- Alt - subtract from the current selection

We could again use the Duplicate menu to get more legs. However, there is even easier way: transformation handles support automatic duplication via Ctrl key. Let's try this.

Switch to the side view. Then press the Ctrl key down and drag the translation handle. The selected legs are instantly duplicated and the move operation is applied to them. Move the two new legs to the opposite edge of the tabletop.



Legs duplicated and translated

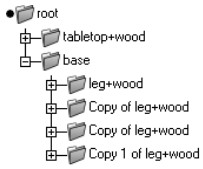
We have now created all the geometric objects we need for our table. Enable Perspective view and OpenGL shading again. Use the same controls as before - they toggle the mentioned View properties on and off.

Manipulating the Hierarchy

We have created all the table parts under the root level. It would be much better to put the legs inside a single dedicated level object 'base'. Such a structure has many advantages. For example, we can then modify all the four legs easily, simply by modifying the 'base' object.

To create a new level, open the popup menu of the Select window with a right mouse button click. Select New/Level from the popup menu. Rename the created new level object as 'base'.

Then multi select the four legs and drag & drop them inside the base level.



Legs moved inside a new level

In addition to drag & drop, you can group objects into new levels using the 'Drop to a level' tool. To move both the tabletop and the base in a separate level called 'table', multi select the objects and then select the item Drop to level from the pop-up menu of the select window. Rename the new level which now contains all modeled parts as 'table'. The root object nown consists of only one sub object, the table.

Creating Light Sources

We now have a wooden table, but the scene is quite dark. Only the dim flashlight of the View camera illuminates the objects. The next task is to add a light source to the scene.

Select the Light Sources section of the Toolbar at the top of the screen and click the Point light source icon.



Activate the point light tool

Switch to the side view and enter two points in the view window - somewhere above the table - to define the position and radius for the point light object. The distance between the two clicks will define the falloff rate of light intensity.

Now, there is also a Point light object in the hierarchy. The tool control bar displays some useful controls for it. Let's experiment with them. For example, change the Falloff property from the default value None to Distance, so that light intensity decreases by distance just like in the real world. Then increase the Intensity value to 1.0.



Note

Unlike the material mapping objects, which affect only objects at the same hierarchy level, light source objects, by default, illuminate all objects in the model regardless of the hierarchy.

Editing Geometry

Earlier, you transformed the tabletop cube using the the transformation handles, which automatically show up when you select an object.

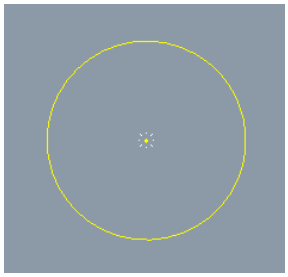
These transformation handles are common to all objects: for example, the rotate handle consisting of three perpendicular circles can be used for rotating all kinds of objects.

Some handles are object specific. You can ask the selected object to show its object specific handles by clicking the Edit button in the Control Bar.

We created the light source by entering two points, which defined the center and the fall-of radius for light source. Both properties can be edited after creation using the Edit mode.

Select the light source and click the Edit button, or simply hit the space bar key, which does the same thing.

A circular handle shows up around the light source. This circle represents the fall-off radius, which specifies the distance where the intensity decreases to 50%. If the light source doesn't seem to illuminate objects of you scene enough, drag this intensity handle bigger.



Point light specific handles

Only the point light source has this kind of handle. For example, a cube defines other kind of handles (six face handles), which can be simpy dragged around to edit the size of the cube.

Ray Tracing

So far, we have worked with our table in shaded mode, so let's see what our table looks like when it is ray traced. Make any adjustments you want to the view orientation using Alt+dragRMB to get a nice view of the table. You can then render the view window by clicking the Render View button, which is at the bottom group of the View Control bar.

Note that all icons have tool tips. If you do not know what a particular button represents, move the cursor over the icon, and in a moment a short message explaining the icon is displayed. You can also activate rendering using the view popup menu Render/Ray Trace.

Shadows and Reflections

Light sources cast shadows by default. In order to see shadows, we need something the table can shadow. So let us use create a floor.

The floor we need can be represented by a large rectangle. Now, you should already have a pretty good idea how to do this: switch the view window to the Top view, use Shift+RMB to zoom out so that you can create quite large floor. Then enable the Rectangle tool in the Analytic tab of the tool bar and enter two points in the view window to define the size for the rectangle. Take a side view and move the floor just below the table.

To make the floor more interesting, you might want to attach a nice material to it too.

Some of the materials define reflectivity, in which case they automatically reflect the surrounding world in a physically correct way. Let's see how this work.

Switch to the front view, then select the Analytic Sphere tool from the Analytic tab of the Toolbar. Click twice in the view window above the table to define the center and the radius for the sphere.

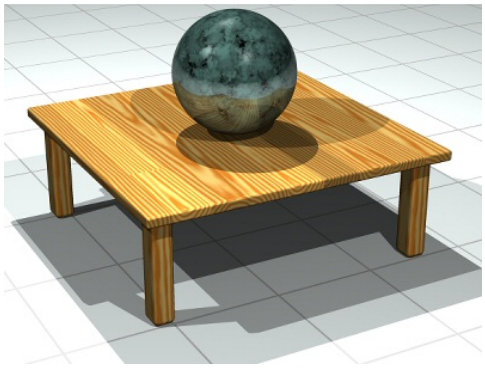
A gray sphere is quite boring, so let's apply a material to the sphere, but this time using the drag&drop interface.

Make sure that the sphere is selected, then switch to Materials tab of the Select window. Move the mouse pointer over the dark 'marble1' material, and press and hold down the left mouse button. You have just "grabbed" the marble material. Now drag the mouse into the view window and release the mouse button. The sphere has been assigned the marble material. To verify this, switch back to the Geometric Objects tab of the Select window and look at your hierarchy.

Drag&Drop is another way to map materials to objects and very useful when you don't have to define a specific mapping geometry. Earlier, when assigning wood to table legs, we wanted to make sure that grains follow the leg direction,and therefore explictly drew the required parallel map. Now we did not worry about the orientation of the marble pattern, and therefore the quickest way to assign a material was quite suitable.

Rotate the view upwards (Alt + dragRMB) and click the Ray Trace button in the View Controls bar to check the result.

The image is starting to look more interesting. The table casts shadows on the floor. The marble sphere should reflect the table.



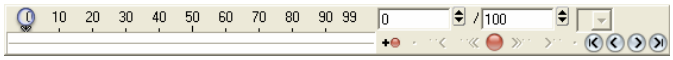
A marbled sphere on a wooden table

Creating an Animation

The next part of our task is to animate the sphere so it rolls across the table. There is much to say about animation, and we will get to that in later tutorials. For now, what you need to know is that you "key" the position, rotation, etc. of objects at particular frames, and let the computer fill in the in-between frames (often called "tweens"). So how do you do it?

Realsoft 3D has the normal main menu below the program title bar. It contains all the general-purpose tools, and if the icon of some tool is not included in the toolbar (because of lack of screen space, etc.), you can find it from here. The main menu also includes the functions that are needed for project management and changing the working environment.

We need to display the Animation window. It appears at the bottom of the main window, below the view window, and looks like the following.

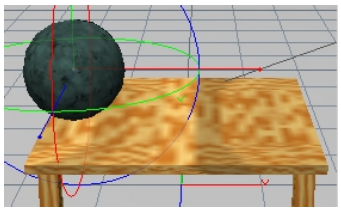


The Animation Window

If this window is not visible already, go to the View menu (main menu bar) and select Animation window from the list. If you don't need animation controls, you can hide the window to save some screen space.

We want to animate the marbled sphere, so select the 'sphere+marble' object from the hierarchy tree of the Select window.

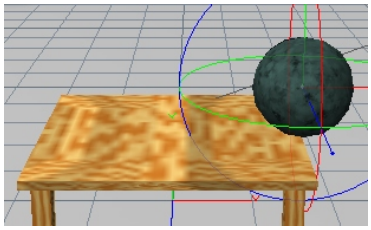
Place the sphere at the position where you want the animation to start (in other words, your initial scene). For example, use the 'x' translation handle to place the sphere at the back of the table.



Sphere in frame 0

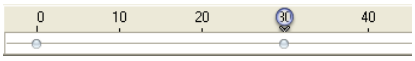
At the right end of the Animation window, there is a red circular button. Click this button. Now animation recording is on and all modifications to the table scene applied while the recording is on are stored as into the animation.

Move the time slider in the Animation window to frame 30, which represents 1 second in NTSC format. If you prefer, move the time slider to frame 25 for 1 second in PAL format.



Sphere in frame 30

Drag the 'x' translation handle to move the sphere to the right edge of the table. Congratulations - you have just created an animation!



Key frames

The Animation Window should now show you two knots, placed at frames 0 and 30 correspondingly. These represent the sphere position in time.

Try moving the time slider between these two knot to verify that the sphere really moves between the two edges of the table when you move the time slider between the two knots in the time line.

You can move these knots to adjust timing. For example, if the sphere moves too slowly on the table, drag the second keyframe from the frame 30 to frame 20 so that the sphere will reach the edge of the table in frame 20.

Let's create a few more key frames. Move the time slider to the frame 60, drag the sphere further over the table edge and then drag the 'y' translation handle to move the sphere down to the floor. Move the time slider to the frame 90. Drag the 'y' translation handle to move the sphere up a bit, as if it had bounced from the floor.

You have now created an animation where the sphere moves on the table, then falls down to the floor and bounce up once.

Click the red animation Record button again to turn off the recording mode.

- Click the Play button in the Animation Window to see your animation.

Animation may not look that realistic yet, as only a couple of key frames is not sufficient to represent a realistic bouncing motion. You can fine tune the animation the same way you created it in the first place: enter animation recording mode and modify the sphere in desired frames to fine tune the motions.



Note

You can also modify the animated sphere animation recording mode disabled. For example, if you animate a sphere to bounce on a table, and you then move it down the floor recording turned off, the sphere bounces on the floor. In other words, if you transform the object in animation recording mode, you edit/create key frames. If you modify the object without animation recording on, then the transformation affects uniformly all motions.

Saving Projects

At this point, we must also perform one important step: save the work done so far. Therefore, select File/Save As from the main menu. Choose a suitable folder and file name for saving the project using the browse button. Press OK to finish the Save operation.

Rendering the Animation

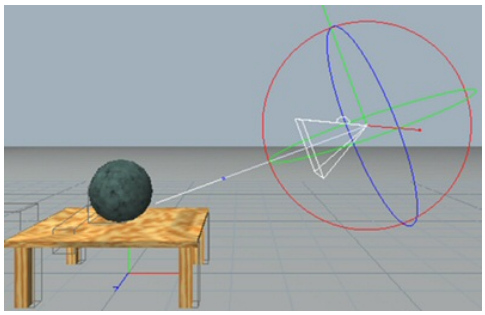
In most cases, the created animation must be rendered to a suitable file format, so that it can be presented as a part of a multimedia show or in a home page, etc. This is the final step for this tutorial.

When rendering images to disk files, it is necessary to create a camera object for the scene. The camera defines how the scene is viewed in the rendered pictures. The view window cannot define this, because there may be many views opened simultaneously, or no views at all.



Click the Camera tool icon on the Creation section of the Toolbar. Then enter three points through the view window to specify the position, the aimpoint and the viewing angle for the camera.

As you can see, the created camera has the standard transformation handles. The camera object is also added to your object hierarchy. It can be selected, modified and animated similar to any geometric object, such as the bouncing sphere above.



Camera object created



Tip

If you click the Accept button in the Control Bar without entering any points in the view window, the orientation of the current view window will be used for determining the orientation of the created camera.

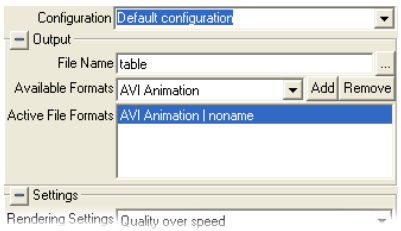
If you want to see how things look from the cameras stand point, simply drag&drop the camera object from the select window into the view window. If you lost your sphere or table in the view window (for example, you accidentally panned the view too much) just drag the desired object from the select window into the view window. This turns the view to show the dropped object.

Adjust the camera position and other attributes until the camera shows the scene in a suitable way.

Select File/Render from the main menu. This opens a File Rendering window with settings for controlling the final rendering of the animation. Do the following:

In the Output section, enter a suitable path and filename (e.g. 'images/myanim') in the File Name field. Use the browse gadget if necessary.

By default the file rendering window renders images and animations to .BMP format, which is not what we want now. Left click to highlight BMP from the Active File Formats list, then press the Remove button to remove the BMP format from the list.



The Rendering Window

Press the downward pointing arrow to the right of the Available Formats field. Select 'AVI Animation' from the list, and then click the Add button.

In the Settings section, select the value 'Reasonable quality' for the Rendering Settings field.

In the Resolution section, set the Width field to 400 and the Height field to 400. The height and width should be the same, because the camera we created above has a symmetric image aspect ratio 1.0 (=square). If the resolution ratio h/w and the image aspect of the camera do not match, rendered pixels will not be perfect squares but stretched rectangles.

Press the Render Animation button to start rendering the scene. This button renders the entire animation, whereas the Render Image button renders a single still image, based on the current position of the time slider.

Wait until the progress indicator shows 100% and the Render Animation button and Render Image button become enabled again. This indicates that rendering has completed. You can now play the animation using your favorite media player. On many systems, all you need to do is double-click the AVI file from the File Manager.

Summary

This first tutorial has introduced several of the basic concepts of Realsoft 3D . An important element of this tutorial was to show you that there are a number of ways to activate various functions in Realsoft 3D.

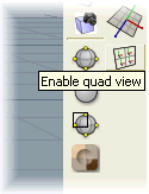
All these different methods of performing the same function may seem confusing at first, but it allows for much flexibility as you learn the program. Beginners may use simple methods like tool icons or menus, whereas experienced users may use fast methods like keyboard shortcuts and compass menus. The program includes thousands of features, and all the input devices - mouse, keyboard, etc. - are loaded with power features. As you learn to use the program, it becomes possible to execute a complicated workflow surprisingly fast by combining these techniques.

This marks the end of the first tutorial. Congratulations on creating your first scene. Hopefully, this first lesson has whetted your appetite, and demonstrated some of the power of Realsoft 3D. Happily, there is much, much more. Further lessons introduce and explore more specific features and techniques. All that can be said at this point is: be patient. 3D modeling is a huge topic, but isn't learning fun!

Modeling Your First Building

Building Walls

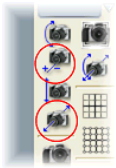
Start the program. Change the view window to the quadview mode by clicking the quadview button. The mentioned button, as well as many other useful view navigation and control tools are grouped into the vertical view control bar at the right side of the view window.



The view control bar is located at the right side of the view window. The Quadview button is highlighted and shows the tool tip.

Click the top left quadrant of the view (labeled as 'Top) to activate it. Then click the Quadview button again to maximize the selected top view. Alternatively, you can press Alt key down and double click the selected view quadrant to maximize it.

Use the navigation controls to zoom out and move the view camera far enough from the origin, so that you will see enough space for modeling a complete building. Note that the status window displays changed camera parameters while you use the navigation controls. The view window should display about at least 15 meters wide area of space, and the camera should be moved at least 10 meters away from the origin. Undesired clipping problems arise, if parts of the building reach out all the way behind the camera.



Zoom and Distance controls



Note

In this software, the XZ plane is the ground plane and Y axis points upwards. You can see the orientation of world from the coordinate system drawn in the view window. The direction convention can be changed using the Space selector of the File/Preferences window.



Activate the Building tool. It is located in the Creation tab of the toolbar.

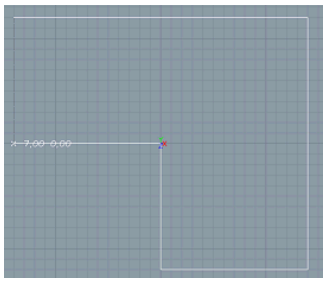
First you need to draw a continuous loop, which defines the outer walls of the house. Click somewhere on the top left corner of the view window to start drawing the walls.



Note

Click the left button - do not drag the mouse while holding the left button down. Dragging operation is reserved for point snapping.

Move the mouse to see a rectangular shape drawn on the view window. By default, the building tool is in automatic mode, which draws a frequently needed rectangular base for the building. However, if the second click is exactly in a vertical or horizontal line from the first click, the tool continues accepting an arbitrary sequence of wall segments. Let's use the latter alternative.



Draw a line sequence, which defines the walls

Move the mouse to the opposite side of the view window horizontally. To easily find an exact horizontal level, just hold the Shift key down. When the first wall segment is 14 meters wide, click second time. Then continue drawing the base of the house until you have an L shaped form shown in the example picture. To finish the drawing, either click the Accept button in the toolbar or click again on top of the first wall point to close the loop.



Note

If you accidentally enter a point in a wrong position, hit the backspace key to step back in drawing.

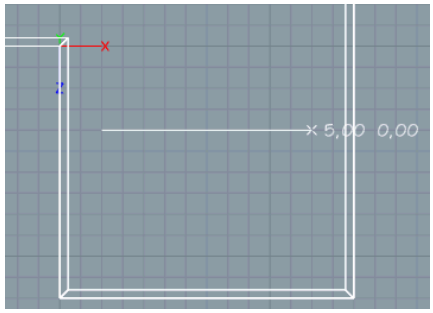
A new object called 'buildingX' appears into the select window, which is located at the left side of the screen. The select window shows you the structure of your current scene. It is very important tool and we will learn more about it later.

Let's draw some inner walls next. While a building object is selected, the second row of the toolbar automatically shows you some frequently needed controls and tools, which are useful for building construction. Click the Sub Wall tool.



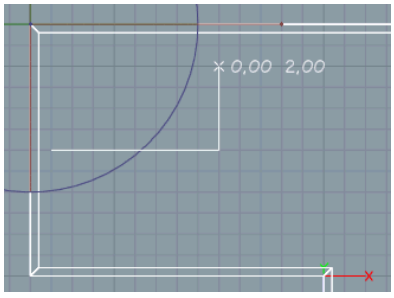
Sub Wall

Click two times to draw a horizontal wall as shown in the picture Note that you do not have to aim carefully to match the inner wall with outer walls. After activating the Sub Wall tool, take a look at the second row of the toolbar; it now shows the options of the Sub Wall tool. Snap Start and Snap End options are active by default, indicating that the tool will automatically fill the gaps. Finish wall drawing by clicking Accept or hitting the Enter key.

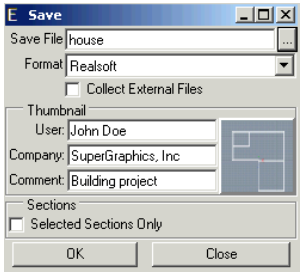


Draw the first sub wall with 2 mouse clicks. Length does not matter.

Activate the Sub Wall tool again and draw a second set of inner walls as shown in the picture. As you can see, sub walls can have multiple points and turns.



The second sub wall has 3 points



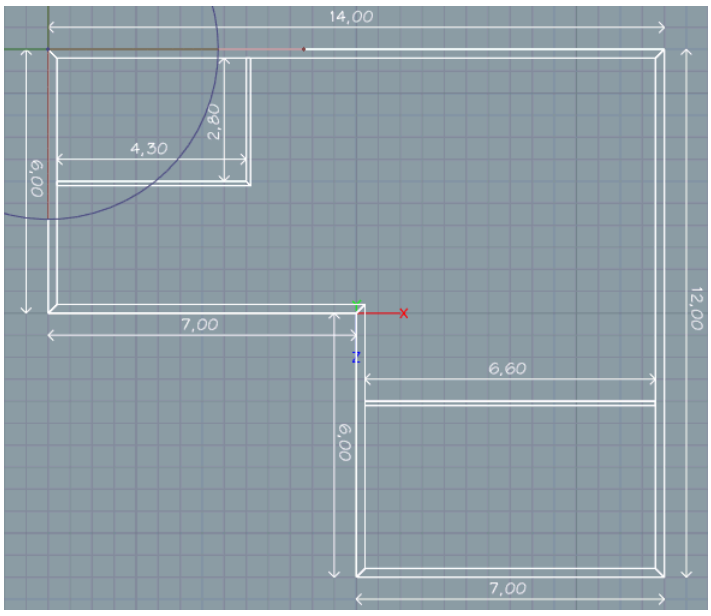
Save the project at regular intervals

It's time to save the work. Select the pull down menu File/Save As. After the save window has opened, define a suitable file name for saving, such as 'house'. You can also fill in some information to the Thumbnail fields. By clicking the thumbnail image, you can record the contents of the view into the thumbnail. Click OK to save the project.

Editing the Walls

The dimensions and walls of the building can be changed easily. All parameters can be found from the hierarchical structure of the building. The building system is fully parametric. When any part of a building definition is modified, the required parts of the geometry are rebuilt again.

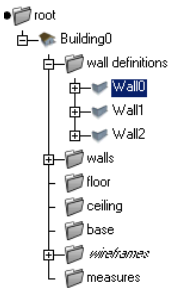
Make sure that the building object is still selected (its name is highlighted in the select window). You can see the dimensions of the walls by selecting Measure Lines=Left from the tool bar.



Measuring lines enabled

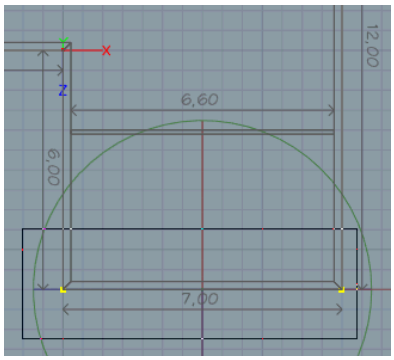
Click the little square which has a + sign inside it at the left side of the building item on the select window. The building folder opens and shows its sub objects. Open the first sub level called wall definitions. It contains three wall objects. The first one is what we drew first, the outer walls. Select it.

Note how the second row of the toolbar immediately shows some options for the selected wall. Change the thickness of the wall from 0.2 meters to 0.25 meters. Then change the extrusion side from Right to Left. The polyline for outer walls originally defined the outer dimensions of the building, but now the walls are built outside the polyline. While you made the changes, sub walls automatically adjusted their length, thanks to the auto snap option.



The hierarchy of the building opened. Outer wall definition selected.

Open the up most wall level on the select window, and select the polyline object called 'nurbsXX' inside it. Hit Spacebar or click the Edit button in the toolbar to put it into the edit mode. The line displays some little red squares - its point edit handles - on the view window. Drag-select (move the mouse holding the left button down) a point pair on the bottom part of the wall, as seen on the picture.



A point pair selected. The drag selection box is also included in the picture.

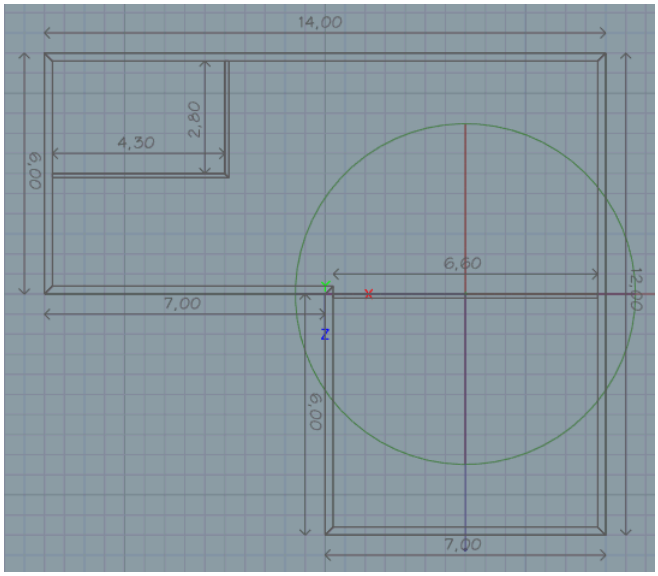
A circle and some axis lines show up around the point pair. The circle is the rotation handle, and lines are translation handles. The end knobs of axis lines are for scaling. Try modifying the point pair by dragging these handles with the mouse. Note that you can cancel an experiment by hitting Esc key, and undo the action afterwards using the Exit/Undo pull down menu. After experiments, expand the building to Z axis direction by dragging the blue axis line 2 grid units downwards.



Note

When you select the nurbs polyline, the toolbar displays a large set of tools for editing the line. With these tools, you can for example add more points or delete existing ones.

Above, we used point editing to modify the walls. A wall can be modified as a complete object, too. Select the second wall object from the select window. Then drag blue axis line on the view window to move the sub wall. Note how the sub wall automatically expands while you move it. Place the wall exactly in the corner as shown in the picture below.



Sub wall moved to a new position

Now we can stop experimenting with wall editing. Close the level 'wall definitions' in the select window to keep the hierarchy display simple. Then select the whole building object from the select window. Turn measuring lines off using the control in the tool bar.

So far, the results of our modeling actions were seen in a simple wireframe model. Next we will experiment how to model in perspective shaded view. But before that, let's add a light source. Alt-double click the view to see the quadview, then Alt-double click the Side view quadrant to maximize it.

The Zoom button is located in the top part of the view control window.

Zoom Tool

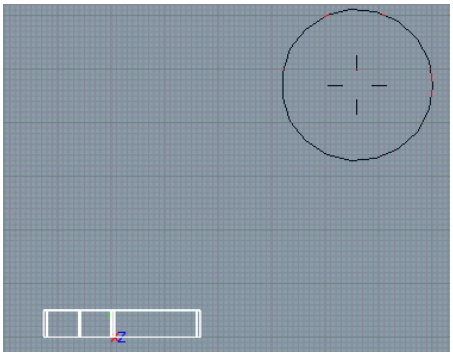
Move the mouse on top of it, press LMB down and drag downwards, so that you will see a larger area of space around the house.



Tip

The quickest way to zoom in and out is to hold the Shift key and the right mouse button down and move the mouse up and down above the view window.

Switch to the Light Sources tab and activate the Point Light tool. Click somewhere above and in front of the house. The falloff radius defined by the second click does not matter this time, so click a second time anywhere on the view.



Adding a point light

Go back to Quadview (Alt-doubleclick) and maximize the bottom right quandrant, which shows the perspective shaded view.

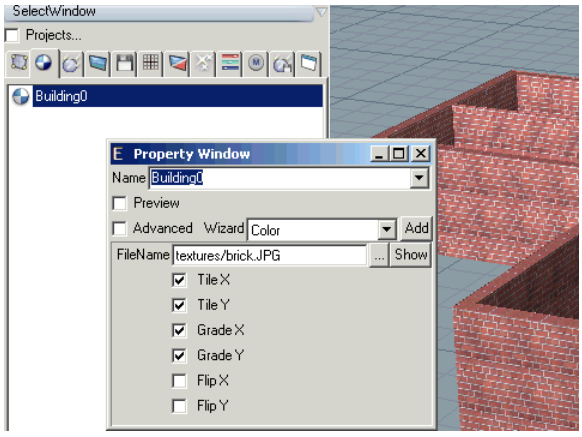
Switch the select window to the second tab from the left. It shows the current material library. The default startup library includes a large collection of materials, but we will need only some of them. So, select Purge from the popup menu of the select window. Only one material called building remains in the material library. Double click it select it and to open the property window.

The building material was actually created by the building tool, when we applied it in the beginning of the tutorial. So called Auto Texture option does this unless the user explicitly turns it off. So, all we need to do now is to click the browse button at the right side of the Filename control, and select a texture map using the file browser. Pick for example bricks.jpg from the textures folder. Then close the property window.



Note

The sub folder 'Walls' of the building contains a texture mapping object 'Default mapping(building)'. It assigns the material to the walls. If you select this material map object, you can change its target material to another alternative using the material selector gadget, which appears in the tool bar as soon as the material map object becomes selected.



Select a brick texture for the walls of the building

Go back to the leftmost tab of the Select Window. Click the building object to select it. Currently the building has only walls. To add a base plate under it, go to the second row of the toolbar and set Base=Fitted to Walls.



Note

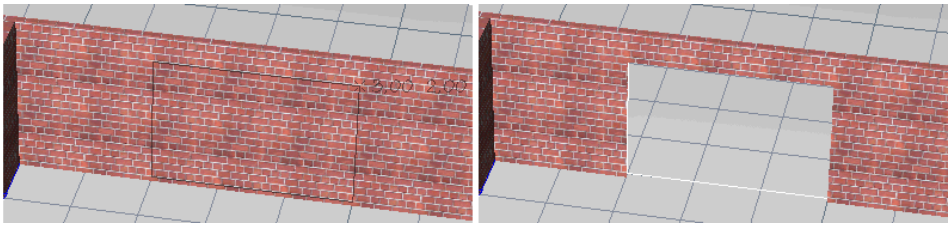
The narrow toolbar has room for only the basic options. For more detailed control, open the Property Window (from the popup menu of the select window, by double clicking the object or using the 'p' hotkey). Then you can adjust the height of the base plate etc.

Rotate the view around the house using the view rotate button in the view control bar. Another way is to hold the Alt key down and drag in the view window with the right mouse button. The view rotates to the direction where you move the mouse. Another useful navigation control of the view control bar is the pan tool (available also via Ctrl+RMB dragging). Try these controls to find a view where you see the straight inner wall.

Let's cut away part of the inner wall. Click the Hole tool in the toolbar.

 *Hole Tool*

Move the mouse pointer to a position where the wall meets the floor and click to define the bottom left corner of the hole. Move the mouse to the left and upward. Note how the hole snaps to the 0.5 meter grid units. Click second time when the hole is 3 meters wide and 2 meters high.



Cutting a wall with the hole tool



Note

Just like the wall definitions, you can find the new hole object from the hierarchy of the building. It is located under a sub level called holes. Open the level and select the hole. Go to the tool bar and change its Width to a new value, say, 2.5 meters. You can also move the hole around using its transformation handles.

It's again time to save the project. Select File/Save As from the pull down menu. Change the file name for example to 'house2'. This way you will get multiple backups of the project, representing different stages of the modeling process.

Windows and Doors

We could add all windows and doors to the building using the hole tool. However, there is even quicker method available: drag and drop from object libraries.

Modeling in shaded perspective view is fun but often the best accuracy is obtained by using parallel projection and a wireframe view. So, alt-double click the view to get back to the quad view and then alt-doubleclick the top view to maximize it.

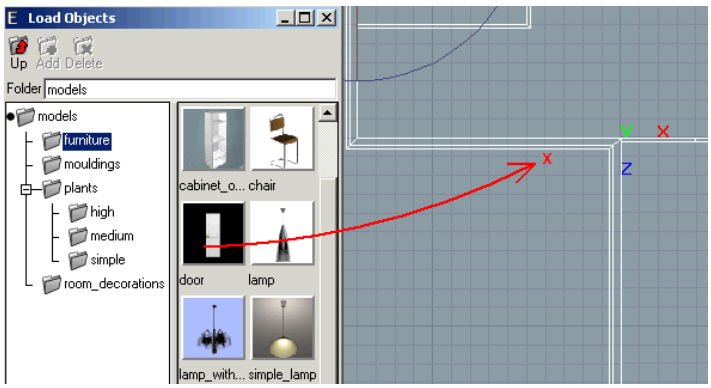
To find a good zoom scale for further modeling, switch to the leftmost geometry tab of the select window, drag the building object and drop it into the view window. Dropping an object to view performs an Auto Focus operation. The view window will zoom exactly around the selected target. You can also use the Auto Focus tool in the view control bar after selecting the building.

 *Auto Focus*

The loaded window and door items should go under the building hierarchy. Therefore, click the 'Building' object in the Select Window. The current level marker jumps from the root to the building level. Then select Browse Object Library from the popup menu. This opens the loader window, which shows disk library objects conveniently as small thumbnails.

Click the library folder called furniture, and find the item door. Drag the door icon over the view window and drop it just outside the point where you want to insert the main door of the building. Do not drop it too far away

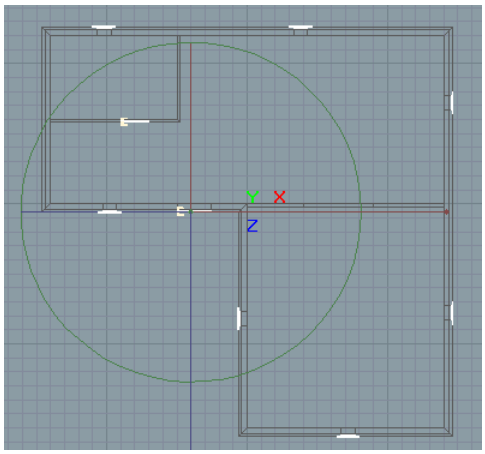
from the wall; the door has auto-snapping properties, which become active if a wall surface is found closer than one meter. If the drop position was correct, the door snaps exactly to the wall and rotates itself to the direction of the wall.



A door snaps to the wall when dropped close enough

The loaded door object appears in the view window and in the select window's hierarchy tree. It is automatically selected and displays useful attributes in the toolbar. Try changing the width to 1.2 meters.

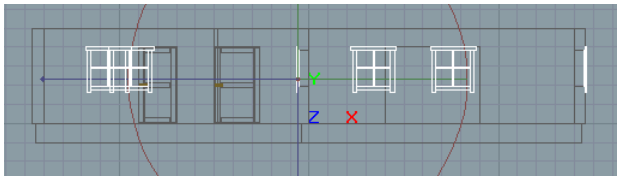
Next drag and drop another door for the small closed room in the upper corner of the house. The width of this door can be smaller, say 0.8 meters - change it using the toolbar control. Then drag and drop several windows so that every wall and room gets at least one window. Note that you have to drop the windows outside the building; otherwise the outer mouldings of the window may rotate to the wrong side of the wall.



Two doors and seven windows added

Close the loader window and take a front view. The windows are quite small; let's make them bigger. In the select window, drag a selection box around all objects called window. You can also select them all by clicking them one by one holding down the Shift key, which is the multi selection modifier.

The windows are parametric objects, which automatically display the dimensions in the tool bar. Change width and height to 1.2 meters. If the upper edge of the windows got too high, simply drag the green translation axis downwards on the view window. Note that holding down the Shift key releases grid snapping temporarily and allows you to align the windows with the upper edge of the door.



Windows resized and aligned with the door

Before proceeding into other steps of building construction, let's clean up the hierarchy. Multi select all doors and windows, and select Drop to a level from the popup menu of the select window. The selected objects are placed under a new folder. Use the Rename tool of the popup menu to rename the folder as doors_and_windows and close the folder. This action keeps your scene hierarchy more compact and better manageable. You can naturally later open the folder, select and edit any door or window if you like. Then save the project with a new name.

Decorating Rooms

Now all walls are made of solid brick, and the floor is just the base plate under the building. The room object allows individual decoration of rooms. This modeling stage happens best in the shaded perspective view. Turn it on as before and rotate the view so that you see the building from above.

Select the building object. As usual, some useful tools for this type of object appear on the toolbar. Click the Room tool button

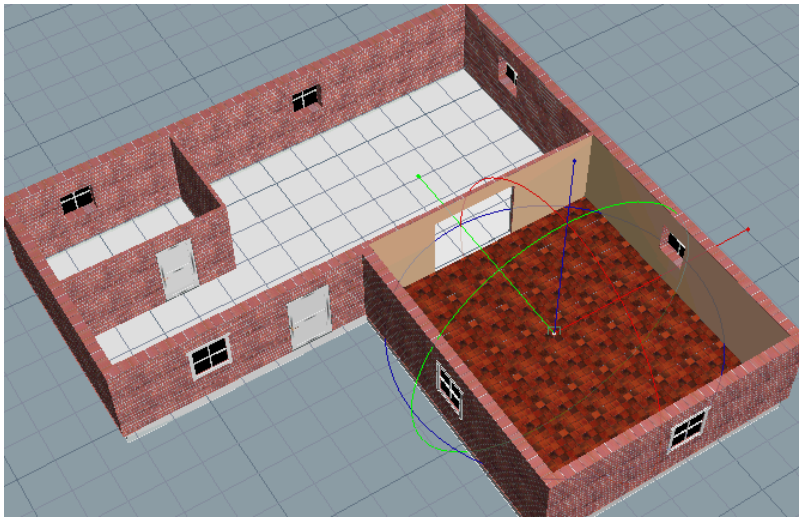


Room Tool

When the room tool is activated, the tool bar shows some tool options. Change the name of the room to 'kitchen'. Click in the middle of the room, which is located in the lower right end of the house. The tool adds a new room object into the hierarchy, and white surfaces enclose the space around the clicked point. Select the room object from hierarchy, and turn off Ceiling option from the tool bar. Now we can see again into the room.

Go to the second tab of the select window, which includes the material library. The room tool has inserted a set of new material templates. Double click the material building_kitchen_floor to open the property window. Open the file selector using the filename browse button and select a texture, for example plates.jpg.

Select a second material, building_kitchen_walls. All surfaces need not to have a pattern. The Wizard selector in the property window shows Color; just click the Add button and a color selector appears on the bottom of the property window. Pick a constant color for walls, for example some nice shade of light brown.



One room decorated

Switch back to the geometry tab of the select window. Open the hierarchy of the kitchen folder. Select a folder called floormouldingsample. Apply Make Current from the select window's popup menu to make it the current level. Then pick the popup menu Browse Object Library. The loader window, which we already used for doors and windows opens. Select the mouldings folder from the browser window. Then drag and drop the item floormoulding_pine somewhere in the view window. Drop position does not matter, as long as it is in the view window. A moulding made of pinewood appears.

Then select the folder ceilingmouldingsample from the select window and apply Make Current popup tool to it. Then drag and drop the item ceilingmoulding_birch from the browser window to the view. Now the decoration of the first room is ready. Select the kitchen object. Just for fun, drag the kitchen object around using the axis handles in the view window. As soon as the indicator reaches another room, the selected surface materials spread around the new room! Easy and fun, isn't it! Finally put the decoration back to its original place and check the Ceiling option in the tool bar. We will leave the ceiling plain white this time.



Note

Instead of loading some predefined mouldings, you can model the profile of the moulding yourself using the toolkit of the Nurbs tab of the toolbar.

We will decorate another room using a method that is even quicker and easier. Select the building object from the select window. Rotate the view so that you can see inside the little corner room. In the object browser window, select the folder called room_decorations. Drag the item 'bathroom' and drop it inside the little room. The room gets fully decorated right away. After viewing the result, just check the Ceiling option in the tool bar to add the top surface to the bathroom. It was initially disabled so that you could immediately check the result after loading the decoration.

If you look at the object hierarchy, you can see that the dropped bathroom object is placed into the building folder. If you do not like the decoration it defines or want to try another alternative, just delete the object.

Decorate the third room either step by step or using predefined decorations. Note that you can easily customize predefined library decorations by changing the textures or mouldings. When all rooms are decorated, close the browser window. You can also close the room object folders in the select window hierarchy to hide the details, which are no longer under our attention. Save the project.

The Roof

To add a roof above the building, select the building object and click the Roof tool in the toolbar.



Roof Tool

By default, the roof has a rectangular base shape and a pyramid like structure. The shaded view shows that again, all surfaces are plain white by default. Switch to the material tab of the select window and double click the material building_roof. After the property window has opened, use the file browser to select a texture map called roof1.jpg.



Textured roof

The roof properties, such as the height, width of eaves etc. can be adjusted from the tool control bar and in more detail using the property window. The actual roof shape can be changed using the Roofline tool.

Go to the unshaded top view. Make sure that the roof object is selected and click the roof line tool.



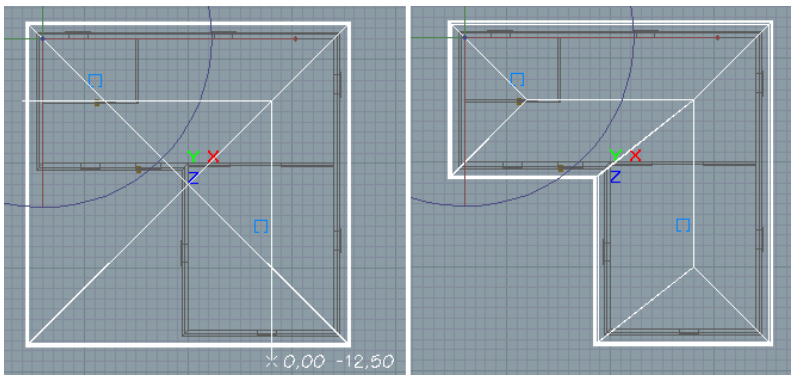
Roof Line tool

Click three times to draw a line that travels in the middle of the L-shape of the building. It does not matter how far the lines reach outside the building. After the third point, click Accept or hit Enter.



Note

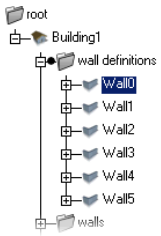
You can use Shift, x and y modifier keys to easily define exact horizontal and vertical lines.



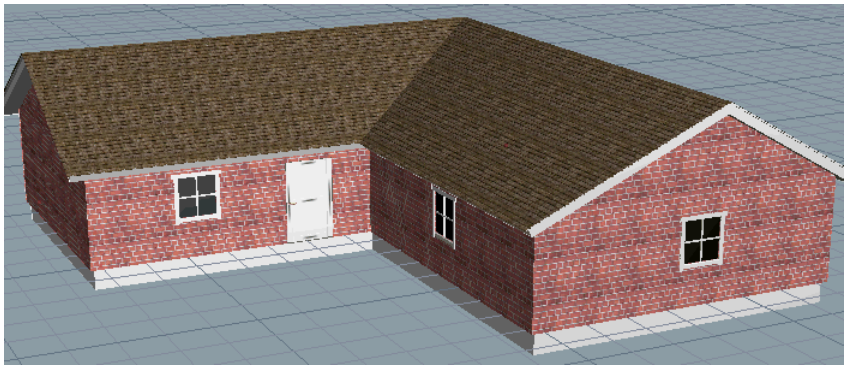
Defining the roof line

The building now has a hip roof. Change it to a gable roof by selecting Roof End = Straight from the tool bar.

Switch back to shaded perspective view and rotate around the house. As you can see, there are triangular holes below the roof at both ends of the house. To fix that, open the wall definitions folder under the building. Select the first wall object (it defines the outer walls). Then activate the Extend to Roof option from the tool bar. The roof is now ready.



Select the outer wall object



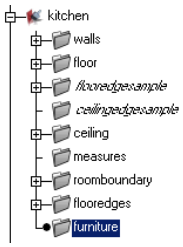
The gable roof finished

Furniture

Furniture can be modeled manually using the general modeling tools. The easiest way is to use the intelligent drag and drop tools and predefined furniture libraries.

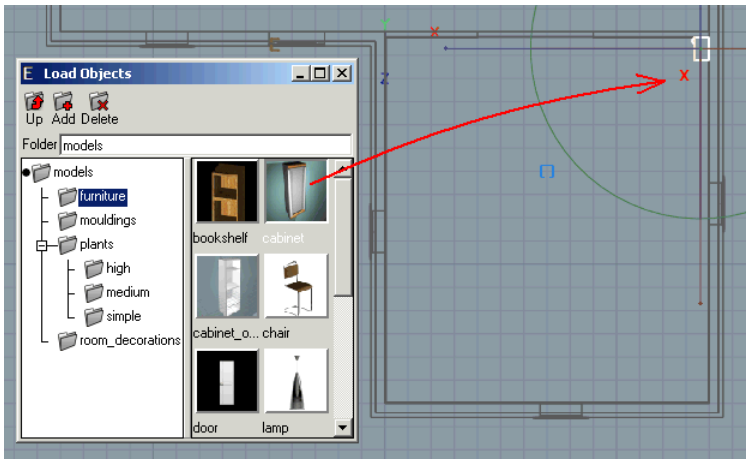
Take the top view (OpenGL shading switched off). Let's handle the kitchen first. Zoom in so that the kitchen fills the view window. Select kitchen from the select window and open its hierarchy, if not open yet. Make sure the kitchen is the current level; note that a small indicator at the left side of the hierarchy level shows what the current level is.

To keep the scene hierarchy in order, all pieces of furniture will be collected to a new level. Select New/Level from the popup menu of the select window. A new hierarchy folder appears to the end of the sub object list of the kitchen. Rename the level as 'furniture'. Then make the level current so that all loaded items will be placed there automatically.



A new level for furniture added into the kitchen and made current

Pick Browse Object Library from the popup menu of the select window. Click the furniture folder in the browser window. Find the item called cabinet. Drag and drop it to the top right corner of the kitchen, 2 grid units down from the corner and one grid unit away from the wall. The drop position is marked with a red cross in the picture below. If the drop position was correct, the cabinet snaps into the corner and aligns with the back wall of the kitchen, which was closest to the drop position.



Drop position of the cabinet and the loaded object. The cabinet aligns itself with the closest wall.

Drag the same cabinet and drop to the same position - 2 grid units down and 1 unit away from the wall. The new cabinet snaps to the end of the first cabinet.

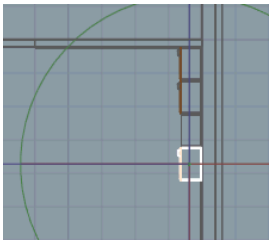


Note

Do not drop the cabinet inside the wall. It may align to the wrong side of the wallpaper!

Check the tool bar after you dropped the second cabinet. It is a parametric object, and you can easily change its width from 0.4 meters to, say, 0.6 meters.

Next drop the cabinet_open to the end of the cabinet row. This time you must shift the drag position one grid unit further. Then add one more cabinet to finish the cabinet group.



Four cabinets loaded



Note

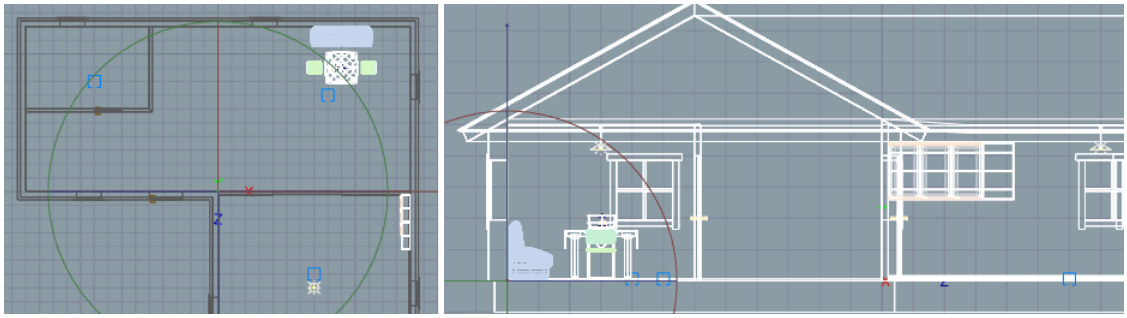
If you accidentally drop a cabinet to a wrong place, just move the cabinet to its intended place or apply Edit/Undo pull down menu and try again.

To finish the experiments in the kitchen, drop the object simple_lamp to the middle of the kitchen. Lamp does not snap to any wall, but it moves in view's orthogonal direction to find the ceiling. If you forgot to enable Ceiling after decorating the kitchen, lamp will travel all the way up to the roof.

Add also some furniture to the middle room. First make it the current level, then add a new sub level called furniture, then make the furniture level the current level. Add the furniture as follows:

- Drop a sofa object close to a wall so that it snaps to the wall.
- Drop a table object in front of the sofa. The table snaps only in Y direction to the floor.
- Drop a chair by the table. Rotate it with the circular rotation handle if necessary.
- Ctrl-drag the chair to the opposite side of the table. Dragging an object with Ctrl key automatically duplicates the object. Rotate the copy 180 degrees.
- Drop a lamp object on top of the table.

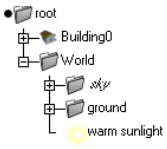
After these steps, take a side view to verify that all objects have snapped vertically to a correct height.



Some pieces of furniture added. Top and side view.

Landscape

The house is floating in empty space. You can model and texture the environment manually, or use predefined world models. We will now use the latter approach. Select the pull down menu Layers/World/flat_ground. New objects appear into the select window hierarchy. The loaded world model also includes a sun, so select the original light point from the select window and hit Del key to delete it.

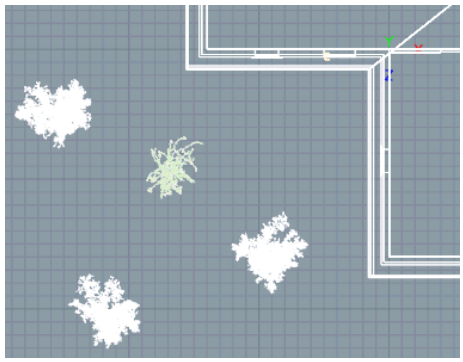


Add a world around the house

Take the front view. Make sure that the base of the house meets the ground level. If not, you can fix it easily:

- Method 1: Select the ground level and move it upwards using the translation handle (hold Shift to release grid snapping)
- Method 2: Select the building and move it up or down
- Method 3: Select the building, open the property window, go to the Spec tab and change the height of the base with the numeric slider

Go to the top view. Select the level called World. Make it the current level. From now on, new items will be loaded to the world level. Zoom out so that you can see the whole building and the surrounding area. We will plant some trees next.



Some trees planted

Open the object browser window from the popup menu of the select window. Go to the folder models/Plants/simple and drag and drop an object called flowertree in front of the house. Then drop three camelia trees to the front yard.

Close the object browser window. The 3 camelia trees are identical, which is quite unnatural. Multi select the trees and click the Randomize tool on the tool bar. Each tree becomes unique.

Close the world level on the select window and click the root level. Then click the Camera tool on the toolbar. First click on the position where you want to place the camera, for example in the front yard. Click second

time where you want to aim the camera - in the middle of the house. The third click defines the opening angle of the camera.

Create a second camera inside the house, for example into the kitchen. Take a front view. Move the cameras upwards to a natural height, say 1.5 meters from the ground level.

Drag and drop the first camera to the view window. The view takes the projection parameters from the dropped camera object. Click the render button, which is located in the lowest icon group of the view control bar. Note that you can scroll the bar with the mouse wheel in case the button is hidden below the visible area. The last step is to save the finished project.



Ray traced image of the house and its environment



Note

You can save the contents of the view to an image file using the view popup menu Render/Save to File. To create print resolution images, use the File/Render pull down menu.

Project Harry

Introduction

In this tutorial, we will model, texture and animate a funny little character called Harry. The tutorial is a complete example and includes all usual steps of making an animation. Although the subject is quite advanced, the workflow is described in detail. Even new users who are already familiar with 3D graphics can try out this tutorial. If you find a certain part of the tutorial difficult, you should examine other beginner level tutorials of the same issue and continue again with improved skills.



Tutorial level: Advanced

Example project: tutorprojects/harry/harry.r3d

Modeling the body

The model will be built using subdivision surface modeling. This technique allows stepwise refinement of a rough control cage into a detailed model. It is a very powerful way to build organic freeform shapes.

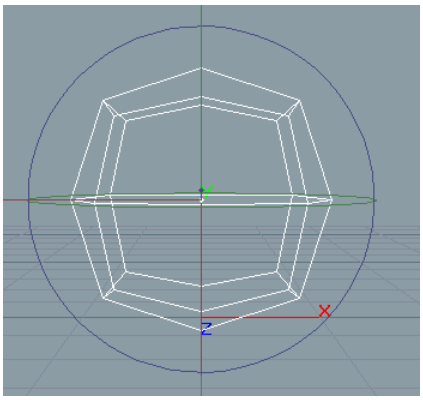
First some terminology: the left mouse button will be abbreviated as LMB, the right button as RMB. Moving the mouse while holding a mouse button down is called dragging. 'Click' means LMB click unless specified otherwise. Similarly, dragging means LMB dragging.

Start Realsoft 3D. You may close the animation bar on the bottom of the screen by hitting the function key F7. This maximizes working space on the view window.

The initial control cage for Harry's body is a simple cube. Switch to the leftmost tab (SDS) of the toolbar. Click on the Cube button to activate the SDS cube tool.

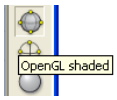

SDS cube tool

Move the mouse over the blue View window. Click once to define one corner of the cube. Press Shift key down to force a symmetric shape (width=height=depth) and move the mouse diagonally away from the first point.


SDS object created

When you see a shape that has a suitable size for the body, click second time. The simple, somewhat rounded shape is the surface generated by a cube like control cage.

Let's modify our modeling environment a bit. To get a shaded real time view of the shape, click the OpenGL rendering icon. It is the third icon from the bottom in the vertical view control bar, located at the right side of the view.

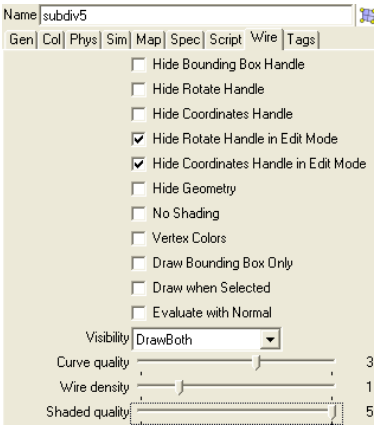


The shaded SDS object may look a bit blocky. To see the actual smooth surface better, double click the object name 'subdivXX' on the leftmost Select window. A quick double click opens the object property sheet (you can also hit a lowercase 'p' hotkey).

Switch to the 'Wire' tab of the opened window. There you can find a slider 'Shaded Quality'. Set it to value 4 or 5, and then close the property window.

In shaded OpenGL modeling, light sources usually help to visualize the shape. Let's add one. Zoom out the view window a bit by holding down both the Shift key and the right mouse button and by moving the mouse downwards. You will now see a larger area of space. Similarly, RMB dragging with Alt rotates the view, and RMB with Ctrl key pans the view.

Fluent space navigation is an important part of modeling in every 3D software package, so it might be a good idea to practice these controls until you feel comfortable with them. If you get lost in space, just hit the blue 'N' icon in the view control bar. It brings you back to the front view towards space origin.



Property Window

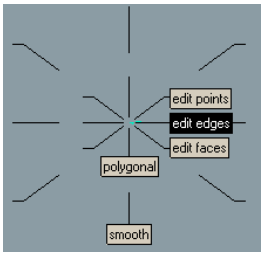

Point light

Then back to the light source issue: the toolbar on the top includes a tab called 'Light Sources'. Bring it to the front. Click the 'Point' light icon. Then click on the view window, for example on the top left corner. Move the mouse a bit and click second time to finish the tool.

After these preliminary steps, we can start the actual control cage editing. Select the SDS object, which we created above, from the leftmost select window by clicking its name (SubdivXX).

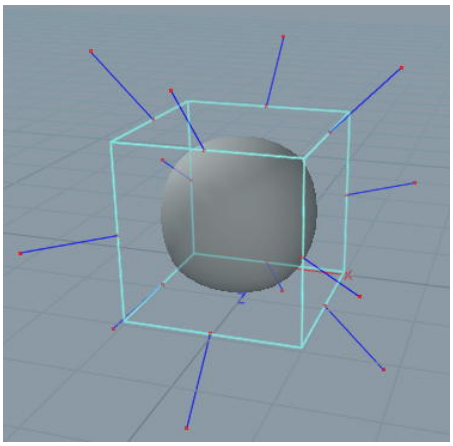
We can quickly set the object into so-called edit modes using the View window's compass menu.

So, move the mouse over the View window, press the right mouse button down and move the mouse. This opens the context sensitive compass menu. When a SDS object is selected, the compass includes among other things 3 edit modes: edit points, edit edges and edit faces. Move the mouse to the right until 'edit edges' becomes highlighted. Then release the right mouse button.

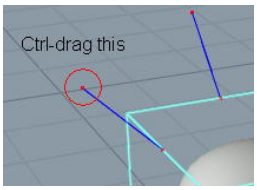


The idea behind the compass menu is that, because it is based on clearly distinctive directions and distances, your hand can learn the selections and make them very quickly. There's no need to aim slowly and precisely.

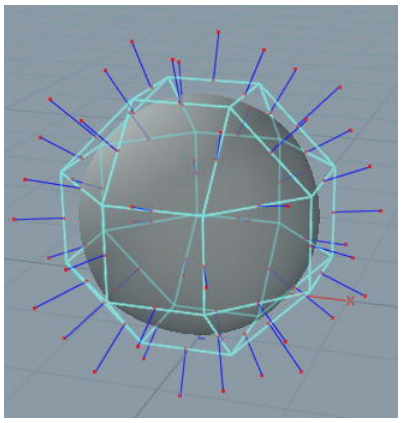
Now the SDS object is in edge edit state. You can see its control cage as a wire frame around the shaded surface. If you move the pointer over the shape, you can see how edge handles become highlighted. Select all edge handles by pressing LMB down and dragging a selection box around the whole SDS object. Selected edges display a blue edge normal handle line. Make sure all the edges are selected. If not, you can add them to selection by Shift clicking them one by one.



Edges selected



Press the Ctrl key down. Move the pointer over a red knob at the outer end of one of the blue edge normal handles. Press LMB down and drag the knob outwards, Ctrl held down. Ctrl dragging an end knob performs the edge bevel operation. Release the LMB when the bevels cut the object into equally sized parts as shown in the figure on the right.



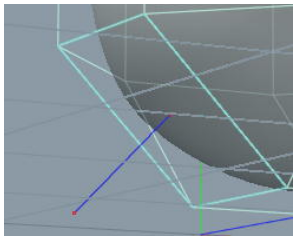
Edges beveled



Note

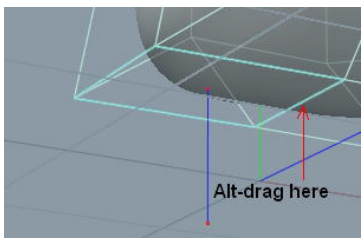
If the operation fails (not all edges were selected etc.), just apply Undo (Ctrl-z) to step back in modeling history.

Now switch to 'Edit Faces' mode; you can do this with the compass menu. Rotate the view (Alt+ RMB drag) so that you see the object from below. Note that the grid on the ground plane helps to visualize rotations. Select a face at the bottom left side by clicking it. This face will be the starting point of a leg.



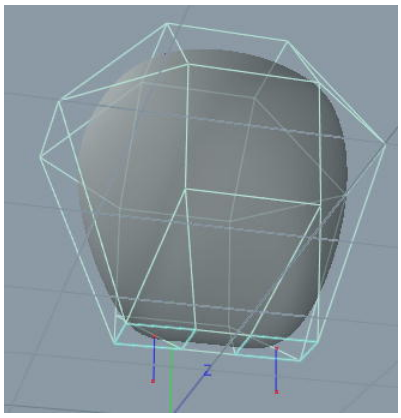
Face selected

Press the Alt key down, move the mouse over the bottom edge of the selected face and LMB drag. This activates rotation of the face around the edge. When the face is fully horizontal, release LMB.



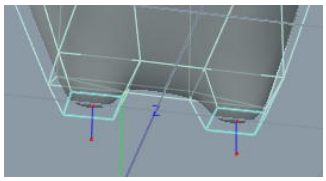
Click the face at the bottom right side to change the face handle selection and rotate it to the horizontal plane, too. Then, press the Shift key down and click the face, which you already rotated. Shift clicking does multi-selection and now you have both faces selected. If you accidentally select wrong faces, you can Alt click them to deselect.

As already indicated, we will 'branch out' two legs from these faces. Before doing that, you may decrease the size of the faces slightly, to make the leg diameter smaller. This happens by LMB dragging the end knob of either of the face normal handles. Note that if you hit a wrong handle, just hit esc key to cancel, or Undo (Ctrl z) afterwards. You may also move the faces closer to each other, by dragging an appropriate face edge. This moves the face in the edge direction. After these adjustments, make sure both faces are selected.

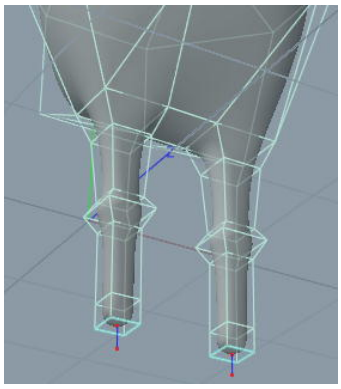


At this point, it is a good idea to save the project. Undo has a limited depth, and you may later want to reload a certain work phase. So, select 'File/Save As' from the pull down menu, type in a file name and hit OK to save. Remember to repeat saving at regular intervals. It is also a good idea to change the file name occasionally so that you will get several backup versions: harry1, harry2, ... etc.

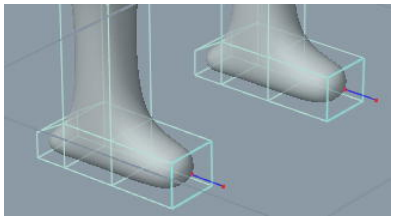
Now we start leg branching. Press Ctrl key down and LMB-drag either of the blue face normal handles from the middle of the line downwards. Two extrusions appear. Leave the first extrusion quite short, so that there will be enough geometry detail to allow leg bending at pelvis area. Release LMB to finish extrusion. Then drag an outer end knob of a blue normal handle line to shrink the leg diameter slightly.



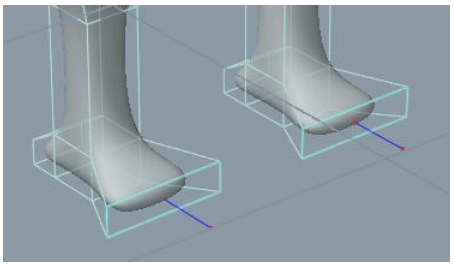
Just keep both end faces of leg extrusions selected and repeat extrusions and diameter adjustments until you reach the heel of legs. Put some extra cross-sections into the knee area, to allow knee bending and better control of surface detail. Feel free to use your imagination here! Learning the keyboard modifiers, such as the Ctrl-drag for extrusion, takes some practicing before it feels easy, but once you learn it, you can change the model this way efficiently. The handles are loaded with all kind of useful functionality, which you can access very quickly.



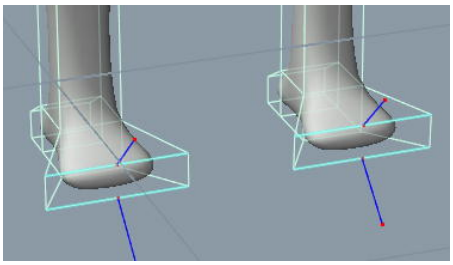
Select the faces behind the leg ends and extrude the heels (Ctrl-drag a normal handle) backwards. Then extrude first foot sections using the two opposite faces. Remember that you can build both legs at the same time, just by keeping the respective face handle selections on both legs.



Increase the foot diameter by dragging the end knob of a face handle. Then make the faces flatter (lower) by Shift-dragging a vertical edge. To summarize edge drag operations we have used so far: plain dragging moves, shift dragging scales and alt-dragging rotates.

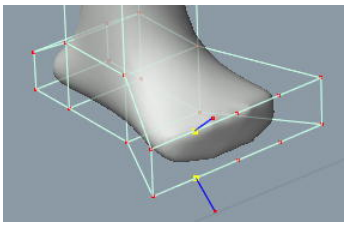


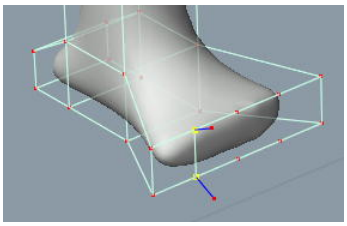
Now its time to extrude some toes. Use the compass menu to switch to the edge edit mode. Click an edge at the top of a foot extrusion. Then Shift click the remaining 3 horizontal edges at the end of the foot extrusions, to add them to the selection.



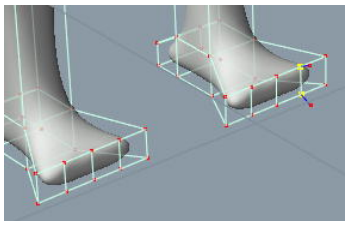
The second row of the toolbar contains a tool icon Split E. Press the icon. Enter 3 to the Splits field and click OK.

Use the compass menu to activate edit points. Click the first split point on the top edge of the left foot to select it, then Shift click the corresponding point below. Now a point pair is selected and you can apply the Add tool of the second toolbar row to create a vertical connecting edge.



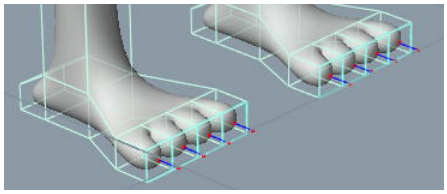


Select the next point pair and apply Add. Do this until all edge point pairs are connected. These steps prepare 4 new faces per foot, each one suitable for branching out a toe for Harry.

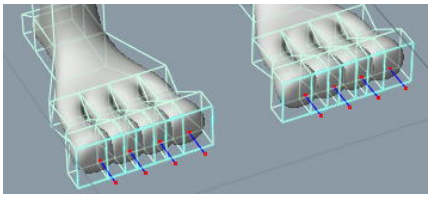


Open the view compass and select Edit faces mode. Click the leftmost base face of a toe to select it. Then shift click the remaining 7 toe faces to multi select them all. Next we will extrude toes.

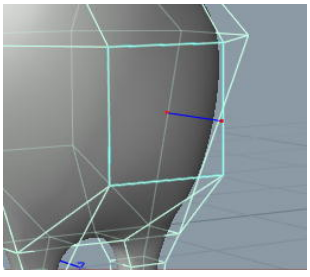
Since we are extruding connected faces, we need to specify some additional options. So, instead of Ctrl-dragging face normals, activate the Extrude tool from the second row of the toolbar. The toolbar changes to show the extrude tool options. Set the Region selector to Separately. Click anywhere on the view window, move the mouse and click again when the toe extrusion length is about one half of the full toe.

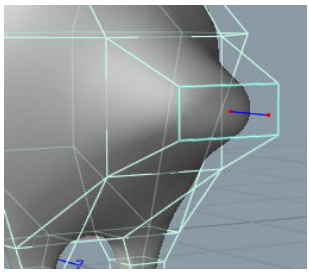


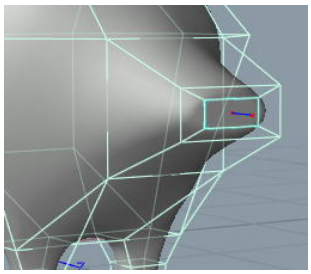
Shrink toe diameters using a red end knob of a face normal, and then extrude a new thicker section out. Again, you can build all 8 toes at the same time. If toes start to overlap, you can move them apart by dragging horizontal edges of selected faces. The final foot is shown in the example image.

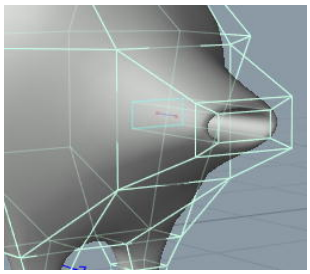


Modeling the beak and mouth takes some simple extrusions as shown in the pictures below. Use the normal handle with the Ctrl modifier to perform them easily. Note that the extrude operation works also inwards - you can create the mouth with it.

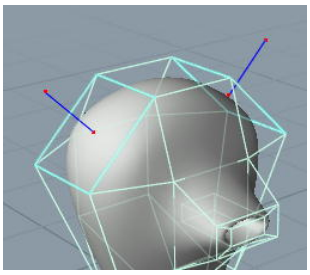




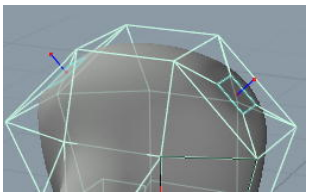




The figures show a very simple mouth construction, but feel free to put some additional geometry detail to the mouth area. This can be done, for example, by splitting the mouth edges and connecting the new points to the surrounding vertices. The more geometry there is, the more flexible expressions the mouth can show.



The two 'horns' or 'eye arms' are extruded from the head. First select the two faces on both sides of the head as the image on the left shows.

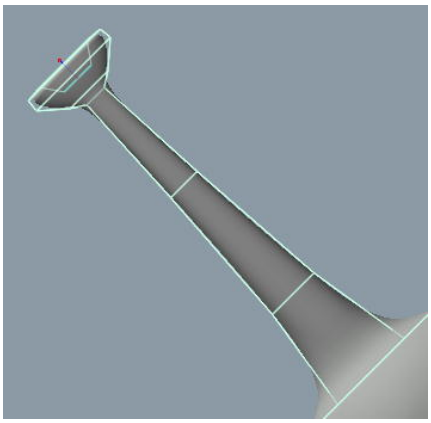


Then press the Ctrl key down and drag the outer end knob of a face normal handle.

Ctrl dragging the end knob does something similar to the extrude operation, except that now the new face appears inside the original one. This way you can set up a smaller area for the actual horn branching.

The remaining modeling steps for horns are just applying extrusion and diameter scaling repeatedly - see the leg modeling instructions above. Add enough cross sections to horns so that you can later bend them in a flexible way and set up interesting poses. One cross section in the middle already allows bending, but with 2 or 3 cross sections more you can achieve multiple 'S' like bends.

At the end of the horns, you should extrude little cup like nests for the eyeballs.



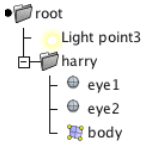
Now the actual body mesh is ready. Fine tune the shape using point, edge and face handles if you like. Then turn edit mode off by selecting 'object' editing from the compass menu (it's the compass selection straight to the left). You can also hit the spacebar to toggle edit mode on/off.

To model the eyeballs, switch to the Analytic tab of the toolbar and activate the Sphere tool. Analytic sphere is a very simple and efficient way to model the eyes.

 *The Analytic Sphere tool*

Hit the blue 'N' icon in the view control bar at the right edge of the screen to get a straight front view. Click in the middle position of the left eyeball. Then move the mouse, and click again when the size is suitable. Then select 'Edit/Duplicate' from the pull down menu, or press Ctrl + d to make a duplicate sphere. Hit 'm' key to activate the move tool. Click the first eye to start moving, move the mouse until the duplicate sphere is located in the proper position in the second horn and click again to finish moving. Take a top view by hitting the camera like icon above the reset icon 'N' in the view control bar. Make sure that the eyeballs are in the middle of the horn also in Z direction. If not, apply the move tool again. Now the eyes are ready.

Let's clean up the hierarchy a bit. Select the body mesh called subdivXX, click the RMB and select Rename from the popup menu. Change the name to 'body' and hit Enter. Select the first sphere and rename it as 'eye1', then rename the second sphere as 'eye2'. Multi-select all three objects (for example: first click the body, then Shift-click eye1 and eye2 on the select window), click the RMB and select the tool Drop to a level from the popup menu. This creates a new hierarchy level, which includes the three objects. Rename the level as 'Harry'. Now we have a clean, simple structure for the character with easily recognizable component names. Organizing and naming the scene parts carefully is a useful modeling practice, and essential in managing complex scenes.



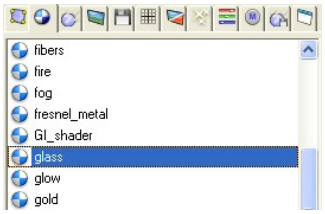
The body is now ready. Save the project again.

Adding Materials

The next step is to apply some materials to the character. You can continue from the previous chapter or start from scratch by loading the example project 'tutorprojects/harry/body.r3d'.

Select both spheres (eye1 and eye2) from the select window; you can do this by dragging a selection box around the two object names, or by Shift clicking them. Keyboard modifiers work on the select window exactly the same way as in the view window. You can also click the spheres in the view window to perform the selection there, although it can be harder there if other objects overlap the desired targets. Note how objects become highlighted at selection.

Switch to the second tab of the select window, which contains a list of materials. By default, material names are displayed, but you can also get their preview icons by selecting 'Show Previews' from the popup menu. Drag and drop a material called 'glass', from the select window into the view window. This assigns the dropped material to the selected objects. Note that object selection, not drop position, defines the targets for material assignment.

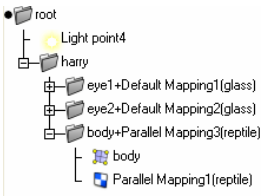


If glass is not in the library, select New/From Template from the popup menu and use the file browser to find and load it in (file 'materials/combined/glass').

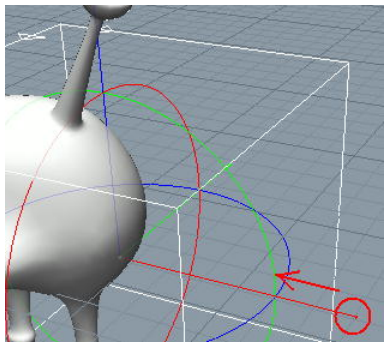
Click the body mesh on the view window to select it. Make sure that the mesh is no longer in Edit mode. If it is, hit the spacebar to turn edit mode off. This is important because dropping a material to a SDS object in edit state may apply the material only locally. Drag and drop a material 'reptile' to assign a bump map to the body.

Let's do a test render. Rotate (Alt-RMB drag) and zoom (Shift-RMB drag) the view to find an appropriate viewing angle. Then click the ray trace icon (on the bottom right corner of the screen) or hit the zero hotkey '0'.

Most likely the bump map pattern, which makes the skin wrinkled, needs some scale adjustments. Perhaps the bumps are too big? Switch to the leftmost geometry tab of the select window. The body mesh has been dropped into a new level with a parallel map object. Select the parallel mapping - it is the object, which attaches the reptile shader to the body geometry.



On the view window, move the mouse pointer on top of the little scale knob at the end of red axis line (other knobs will do as well). Press Shift key down and drag the knob towards the handle center. Release LMB when the map geometry has shrunk sufficiently.



Render the view. Now bumps are smaller. However, they appear much steeper because the height did not change in density scaling. To adjust bump height, switch back to the material library on the select window, double click the Reptile material to open the property window and change the Bump Height value (default 0.01 m = 1 cm) to a smaller value, such as 0.002 = 2 mm. So, type 0.002 in the Bump height box and hit Enter. You can now close the property window.

Now bumpiness is OK but there is one problem: bump pattern is tied to the body geometry along a rigid parameter grid, defined by the parallel map box. The simple box cannot follow when Harry for example bends its leg. Therefore, we must convert parallel map based pattern to a pattern that is fixed to the actual vertices defining the body.

There is a tool for this purpose. On the geometry tab of the select window, select both the body mesh and the parallel mapping. Go to the Materials tab of the toolbar and activate Map2Obj tool. The default options, Target=Points, Source =Map Coords and Destination=UV Coords are suitable. This will evaluate and store parallel map coordinates into every point of the SDS mesh. Hit Accept.

Now you can delete the parallel map object (use Delete hotkey or Edit/Delete pull down menu). Then select the parent level of the body mesh. Switch to the materials tab ob the select window. Make sure that Reptile is still

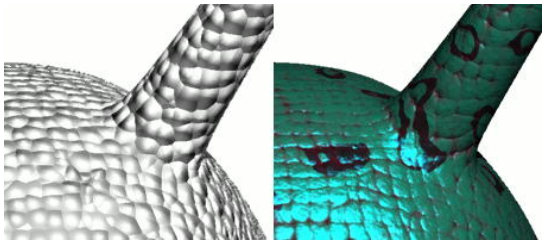
highlighted, and select Map/Default from the popup menu. This creates a new kind of material map, which uses UV coordinates. UV coordinate values are stored into vertices, so they will not change when the model deforms.

Save the project.

Programming the Skin with VSL

The reptile skin material is now attached to the body. It defines only bumpiness, no color patterns or other shading information. Let's make the skin material more interesting:

- More variations to bumps
- Color patterns
- Metallic shine
- Efficient antialiasing



Original and improved skin

The example project to start this chapter is 'tutorprojects/harry/body_and_materials.r3d'. You can also load the improved materials from the example project 'body_and_skin.r3d' if you wish to skip this VSL oriented chapter.

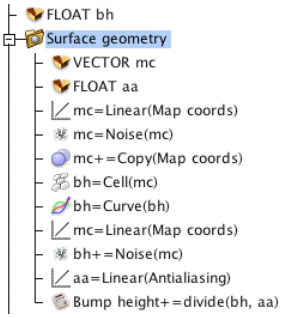
Surface Geometry

Switch to the material tab of the select window, make sure the reptile material is selected and open the property window. Check the Advanced option to see the structure of the material. It is very simple:

```
Surface geometry
    Bump height += Cell (Map coords)
```

A VSL material consist of top level blocks called shaders: Surface geometry, Surface properties etc. Shaders define what happens in each step of the rendering pipeline. As the name suggests, a Surface geometry shader defines geometrical modifications for target surfaces. Although you can add shaders into a material in an arbitrary order, shaders have a fixed, natural order of evaluation. For example, a Surface geometry shader is always evaluated before a Surface illumination shader. Within shader levels, the evaluation order is the order of the objects in the VSL tree.

The first step is to make bumps more interesting. A surface geometry shader that does this looks as follows:



Below is a description how to build the shader. Creating a finished shader from scratch may look like a complicated task especially for users with no programming experience. However, in a real modeling situation, such shaders are created one small step at a time. For example, you could first solve how to control the bump shape only. While adjusting the shader and rendering previews of it, that part of the shader becomes very familiar to you and it is easier to proceed into the next part.

Let's start: drop a Variable object to the root level of the material. You can add VSL objects either by selecting the target level in the VSL hierarchy and choosing a VSL object from the popup menu, or by drag and dropping VSL object icons from the VSL Objects frame. The latter alternative has the advantage that you can place the object exactly at the desired location. Popup menu adds new objects to the end of the selected level, and you often have to drag the new item to the right place.

The new VSL object - Variable - becomes automatically selected. The bottom part of the VSL window shows a property gadget for it. Set the Type of variable to Float and rename it as 'bh'. VSL supports 3 kinds of

variables: one dimensional float and 3 dimensional color and vector variables. The rendering system handles type conversions automatically, but it is wise to use the most appropriate type. For example, when you assign a value to color variable, VSL editor shows you a RGB color selector instead of a numeric vector gadget. Bump height is clearly a one dimensional value, so we use a float variable here.

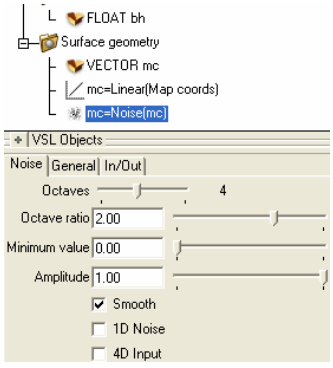
This root level variable 'bh' will store bump height information and pass it to the Surface properties shader - we will use it there to define color from the bumpiness. Root variables are visible in all shaders, and therefore they can be used to pass information between shaders. Adding a new channel is another alternative, but this is a lighter, only locally visible alternative. One must be careful when passing information in root variables. Because of ray trace recursion, the material can be evaluated several times between shading steps. For example, do not expect that a root variable initialized in Surface properties still contains the same value in Surface illumination. Shadow computation may evaluate the same material on another position to find out object's transparency. The method can be used here because there are no shading events between Surface geometry and Surface properties shaders in our scene.

The original bumps were quite regular. A standard solution to make a pattern irregular is to add some noise to the parameter channel. The channel that contains the 3D space related parameter (suitably warped by a geometry of a material map object such as a parallel or a spherical map) is Map coords. Its original value may be needed later, and therefore we compute the modified parameter and store the result to a local variable. So, drop a new variable object to the beginning of the Surface geometry shader. Set the type to Vector and rename it as 'mc'. Add a Linear object after the variable, set its Input0=Surface:Map coords and Output=mc (you can change Input and Output from the popup menu of the shader tree gadget). Set the Multiply value of the Linear object to (0.5 0.5 0.5). This value controls the density of bump shape irregularities. Density should be in a suitable proportion to the bump size. If the noise is clearly denser as the skin cell pattern, individual cells get distorted. If the noise density is lower, cells keep their original shape but their positions change and dimensions twist along the noise curvature.



Original skin, distortion by a low density noise and distortion by a high density noise

Add a Noise object after the Linear object, set Input=mc, Output=mc, Octave ratio=2, Smooth option enabled and Amplitude=0.1. Amplitude is the most important value here - it defines how large the bump shape irregularities become. You can find suitable values by experimenting, but 0.1 is a good starting point. The Octaves value of the noise controls how much detail the noise has. If you plan to make an animation, which shows the skin from a very close distance, you should use a high number of octaves so that the magnified skin shows enough details. For this example, 3 or 4 octaves is enough.



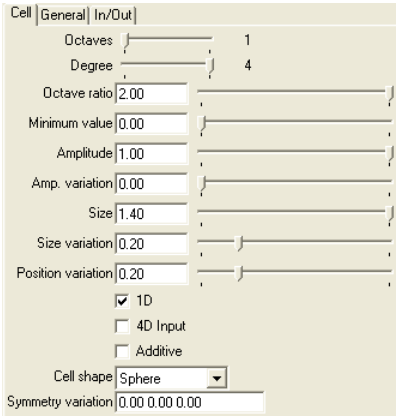
Noise properties

Now mc contains noise values for the bump shape distortion. To get the original parameterization included, add a Copy object after the Noise object, set Destination to mc from the popup menu, Source to Surface:Map coords and Operation from the General tab of the property window to +.

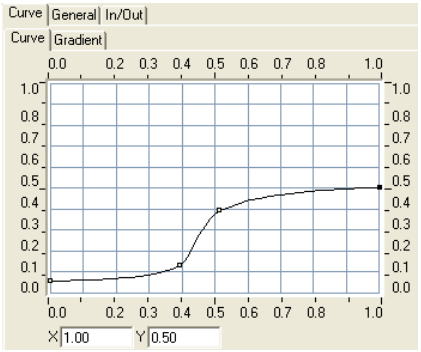
Now mc is a sum of irregular noise and regular Map coords. Using pure noise to define bumps would create a totally distorted bump pattern. Therefore we will use a sum of noise and Map coords. To summarize the computation with some clarifying comments:

```
mc=Linear(Map coords) // Define noise density as mc = 0.5 * Map coords
mc=Noise(mc) // Compute noise value.
mc+=Copy(Map coords) // Add regular parameter
```

Next activate the Cell object, which is already in the shading tree. It should be the last object in the Surface geometry shader. If not, drag it to the end of the object list. Change its Input0 to mc and Output to bh. Change Minimum value to 0 and amplitude to 1. We will modify the bump shape with a curve object, and the 0...1 range is a suitable parameter interval for it. In the General tab, change the operation from '+' to '='. The Cell object now initializes the new bh variable instead of modifying the actual Bump height channel. Other Cell settings can be as in the figure below.



We can adjust the shape of bumps with a curve object. The purpose is to define grooves between relatively flat bumps. Add a Curve object after the Cell object and set Input0=bh and Output=bh. Open the popup menu over the Curve gadget and select Set Minimum and Maximum Values. The default Curve scale, 0-1 meters, is definitely not suitable for skin details. Set Max Y = 0.01 meters in the opened dialog, set Rescale and hit O.K. Then edit the curve as shown in the picture below.

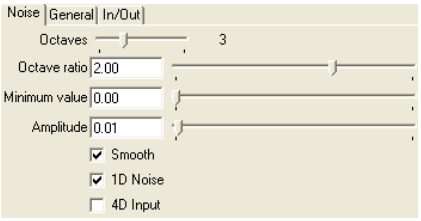


If you want to preview the bump pattern now, just change the Output of the Curve object to Bump height and render. When the result is OK, change Output back to bh to continue the example. The curve controls the profile of bumps, and you can change it here easily and intuitively. Just note that the curve defines only one half of the bump, from the edge to the middle point.



The bumps have now somewhat irregular shape, but the surface does not have very fine detail. The bumps are actually quite smooth. The standard solution is to add a suitably dense noise field. Because the fine details must be clearly denser as 'large scale' bumps, we need again a modified parameter. So, drop a new Linear object to the end of the Surface geometry shader and set its Output to mc. We use the already defined mc variable here again, because its old value is no longer needed. Set Multiply of the Linear object to 5 5 5. This will give roughly 5 times denser bump pattern as the base pattern consisting of cells.

Then add a new Noise object to the end of the shader. This is very fine detail noise, so set Amplitude to 0.002 (2 millimeters) by typing 0.002 in the Amplitude box. Activate the Smooth option (there are no sharp edges on the skin) and set the 1D Noise option, because Bump height is a one-dimensional value. 1D noise is faster to compute as 3D or 4D noise. Set Input0 to mc, Output to bh and Operation from the General tab to '+'.



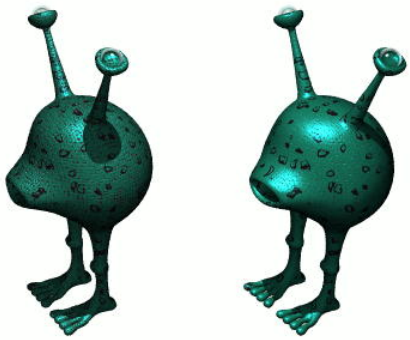
The final step is to move the computed bh value to the actual Bump height channel. We do this in an intelligent way, which improves antialiasing quality efficiently.

First, drop a new variable to the Surface geometry shader. You can place it to the beginning of the shader to make the VLS code nicely structured. Change the type of the variable to Float and rename it, for example, as

'aa'. Then add a new Linear object to the end of the shader, set Input0=Surface:Antialiasing and Output=aa. The Multiply value of the Linear object should be relatively high, 500 or so. The value of the Add field should be 1.

The last VSL object is an Operation object - add it to the end. Set its Type to Divide. Input0 should be bh, Input1 = aa and Output = Surface:Bump height. In the General tab, change the assignment Operation from '=' to '+'. It is a good practice to use + operation when modifying the bump height channel. All materials coded like this can be multi mapped to a same target object and bump maps become automatically summed up.

Let's consider the last VSL line a bit more: the computed bump height value is divided by an antialiasing factor, which describes 'information density' around the examined point. The density is very high if you view the object from very far away or in a low angle. When we divide the bump height with a high-density factor, bumps will gradually fade away. This is exactly what we need: small specular highlights from bump peaks, dark grooves and other details which would cause distracting flickering in animations unless a very high ray trace sampling rate were used, now automatically blend towards a nicely behaving smooth surface.



No antialiasing on the left, shader based antialiasing on the right

The Surface geometry shader is now ready; make some test renders and play with the Multiply value of the Linear object which scales Antialiasing, to find a suitable value, which does not reduce orthogonal bumps close to the camera too much.



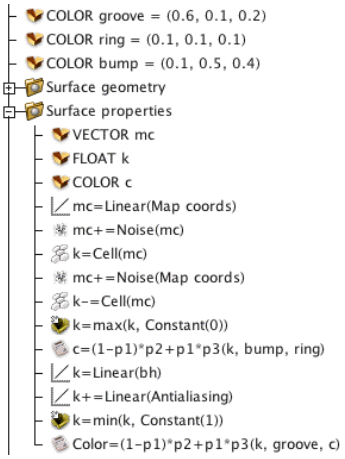
Note

If you intend to use displacement mapping with these antialiased bumps, remember to turn antialiasing off by setting Multiply=0 for the Linear VSL object which scales antialiasing channel. The shader computes a different bump height value for a camera view and for light sources. In other words: light sources see the object shape in another way as the camera does. Inconsistent geometry definition by surface displacement will create strange shadows, which almost certainly spoil the image.

Note for advanced users: the antialiasing channel has a 3 dimensional value. The third Z component contains a base value which depends on camera projection and focal length, distance from the examined point and the surface angle against the viewing ray. The first two sub components X and Y incorporate the contribution of the material mapping geometry. Shrinking a parallel map dimensions to one half doubles the antialiasing X and Y factors. This option is useful for example when mapping bitmap textures to surfaces. In our example, parameter is defined using a Default map object, which simply moves the value of UV coords (stored in vertices) to the Map coords channel. The Default map has no 3D space geometry of its own, and therefore it sets all three sub values of the antialiasing channel to equal. In the example above, we could use any of the sub channels; now the first X component is used because of the type conversion rules of VSL.

Surface Properties

Next we will define some color patterns for the skin. The goal is: bumps have a certain base color and the grooves between them have a different color. In addition to this, bumps have some large dark irregular rings on them. Rings do not color grooves.



Add three color variables to the root level of the material and rename them as groove, bump and ring. For each variable, check the Initialize option and set the color value as you find suitable.

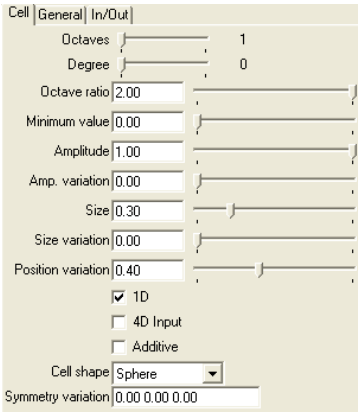
In the example scene, grooves are dark bluish red (0.6 0.1 0.2). Bumps are turquoise green (0.1 0.5 0.4) and rings are almost black (0.1 0.1 0.1). Initializing the values into root variables is an efficient solution: it is done only once at the beginning of rendering, not in every examined point. You can also initialize the values efficiently using Constant objects in a Material Initialization shader (bump = Constant(0.1 0.5 0.4) etc.).

Add a new shader object to the material. Its default type, Surface properties, is just what we need. First we compute a value, which adds the rings. Add two variables to the beginning of the Surface properties shader. The first one is a vector variable 'mc' and the second is a float variable 'k'. Set the types and names as before. The variable mc is used the same way as in the Surface geometry shader: we compute a noise distorted parameter into it, so that we can generate irregular patterns. Computation goes now as:

```
mc = Linear(Map coords)
mc += Noise(mc)
```

So drop a Linear and a Noise object to the Surface properties shader, and set Input and Output channels as shown above. The Multiply factor of the Linear object should be now much smaller, for example (0.2 0.2 0.2). The variable mc will be used for ring patterns, which are much larger as the skin bumps. When we multiply the Map coords with a number smaller than one, the product changes slower in 3D space and hence the patterns become larger. Noise Amplitude can be for example 0.07. The bigger the value, the more irregular rings you will get. Change also the assignment operation of the Noise from = to + in the General tab.

Next add a Cell object to the end of the Surface properties shader. It defines where rings appear. Set Output=k and Input0=mc. Cell options are shown in the image below. It is important that Minimum value is 0 and amplitude is 1, and amplitude variation is 0 - we will use k variable to 'key' the black rings to the skin. Also, cell Degree must be 0 so that the output value jumps straight up to 1 from the base value 0. A higher degree would produce gradient-edged rings. Set also 1D option for faster computation. Size variation can be 0.



The Cell object now produces spots, not rings. Subtracting a slightly modified smaller Cell signal turns spots into rings. To modify the inner spot shape, we add more noise to mc. So, drop a new Noise object to the end of Surface properties. Set its Output to mc and Operation to +:

```
mc += Noise(Map coords)
```

Noise Amplitude must be quite small, for example 0.002 - otherwise rings will easily break. Press Ctrl key down and drag the already added Cell object to the end of the Surface properties shader. This creates a duplicate of the Cell object. Decrease the Size from 0.3 to 0.2. Change the Operation from = to -.

To ensure that the key value k does not get negative, drop a Constant object to the end of the Surface properties shader. Set its Output to k. Change the Operation to max:

```
k = max(k, Constant(0))
```

Now the key value is set. We need one more color variable. Drop a Variable object to the beginning of the Surface properties shader and rename it as 'c' (you can also use longer names if you like). Add an Operation object to the end of the Surface properties shader. Set the Type of the operation to the formula labeled as (1-p1)*p2 + p1*p3. This cryptic looking formula actually does simple linear interpolation. With such an operation, you can blend a color over another color using 'alpha key'. Set Output to the color variable c, Input0 to k, Input1 to bump and Input2 to ring. Remember that the two last input parameters were the colors defined in the root level.

Finally we will key color c - combination of bump and ring colors - with the groove color using the bump height as a parameter. In the Surface geometry shader, we stored non-antialiased bump height to a root level variable bh. Drop a new Linear object to the end, set its Output to k and Input0 to bh:

```
k = Linear(bh)
```

Because the bump height was maximally about 0.005 meters, the Multiply value of the Linear object should be 200. That scaling gives us the suitable keying range 0...1.

We can antialiase also the color pattern efficiently within the shader, just like we did with the bump height. The idea is to blend the color pattern towards the average color. Especially fine details, such as the thin grooves between bumps, need careful antialiasing. We can incorporate the antialiasing step easily into the final color interpolation as follows: drop a new Linear object to the end of the Surface properties shader. Set Output=k, Input0=Surface:Antialiasing and Multiply=100. The Multiply value may need some adjusting later - this is just a guess to get started. Set Operation in the General tab to +. Now the interpolation key value grows by the bump height and by the antialias channel value. Therefore, the groove color will gradually fade away at object edges or when the object gets further away from the camera.

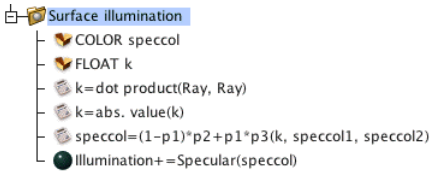
To ensure that the key value - a sum of two factors - does not exceed 1, add a Constant object to the end. Set its output to k and the Constant value to 1.0, and change the General tab's Operation to 'min':

```
k = min(k, constant(1))
```

The last shading action in Surface properties is color interpolation. Ctrl -drag a copy of the already existing Operation object, which computes c, to the end of the shader. Change the Output to Surface:Color, Input1 to groove and Input2 to c. The definition of surface's color is now ready.

Customized Specular Illumination

The third shader will control specular illumination. A metallic shine is very easy to add, so let's make the shine more interesting by controlling its color with a viewing angle dependency. Shine color will change from green to blue, depending on how we look at the surface.



Add two new root level variables to the beginning of the material, where you already placed some other root variables. Both new variables are color variables. Rename the first as speccol1 and the second speccol2. Check Initialize option for both variables and choose the color you like; in our example scene, speccol1 is light green (0.1 1 0.5) and speccol2 light blue (0.1 0.5 1).

Select Specular Color from the wizard selector at the top of the VSL window and hit the Add button. This initializes the illumination shader, which we will customize. Add two variables to the beginning of the new Surface illumination shader. Rename the first color variable as speccol. Change the second variable to type Float and rename it as k. The latter variable is again an interpolation key.

Drop a new Operation object to the Surface illumination shader, just above the last Specular object. Change the operation Type to dot product. Set Output=k, Input0=Light:Ray and Input1=Surface:Ray. Dot product is a quick way to measure angle between two directions: k=0 when rays are orthogonal, -1 or 1 when rays have opposite or parallel direction.

To ensure that the key is positive, drop another Operation object after the previous one. Set the Type to Abs.value, Input0=k and Output=k. Now k is suitable for interpolation. Add a third Operation object above the Specular object and select the interpolation formula (1-p1)*p2+p1*p3 to the Type field. Set Output to speccol, Input0 to k, Input1 to speccol1 and Input2 to speccol2. This operation will interpolate a specular highlight color using the surface viewing angle from two base colors. The result, varying metallic shine, will look interesting especially in animations. As the final step, select the last Specular object and set its Input0 to speccol.

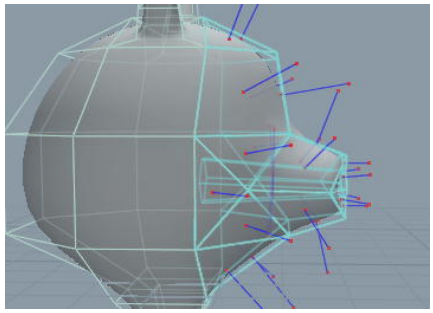
At this point, you should save the project again.

Local UV Texture

The body has now a procedurally defined color texture that covers the surface everywhere. This is a good starting point. However, details, which follow the geometry in a unique, irregular way - for example color bands around the mouth - can be hard to add procedurally. The easiest solution is to use a painted texture, which is wrapped onto the face of the character. We will do this next using so called UV Set object, which defines the texture only locally.

Start situation for this chapter: tutorprojects/harry/body/and/skin.r3d

Take a side view. Select the body mesh. In the screen shots below, we have turned off OpenGL shading of the skin textures. OpenGL visualization will not be accurate enough for estimating texture continuity. Also, when later pasting hair and animating the character, non-textured body renders much faster in OpenGL. To turn off the skin textures, open the property window, go to the Wire tab and set Texture quality to None.



Put the SDS mesh to the face edit mode. Activate all SDS faces at the face area of the body.

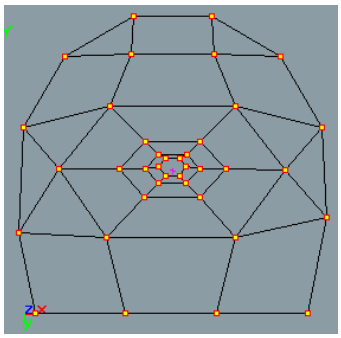
The tool control bar includes a tool called UV Set. It is at the right edge of the control bar, so you may have to Ctrl-RMB drag the control bar to find it. Click the tool. A new item 'SDS UV Set' appears to the hierarchy.



UV Set created

By default, this UV set object takes the UV coordinates from the parent SDS mesh. Those were defined using a parallel projection, which we adjusted for the reptile skin. Such UV coordinates are probably not suitable for mapping a texture image to the face.

To improve this, take a front view (=view against the UV set area), zoom the view so that the UV set fills the view area and click the From View tool (while SDS UV Set is selected) in the tool control bar. This tool uses the current view projection to define UV coordinates.

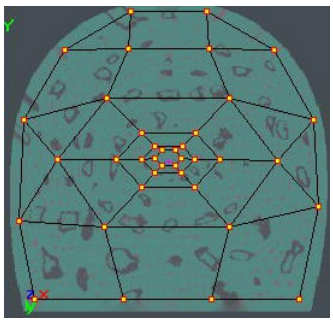


Open the UV editor from the pull down menu Windows/UV Editor. You can now see the defined UV mesh; hit the spacebar to set it to the edit mode. If the object is in face edit mode, choose Edit Points from the UV Editor's pop-up menu. Point edit vertices to adjust the result. For example, the mouth area may contain overlapping points - move them apart. The Unwrap UV tool can be helpful in fixing dense areas and folded surfaces. Close the UV editor when UV mesh is good enough.

We can 'bake' the existing color pattern to a texture map. This will make it easier to join the painted area to other skin. To initialize the texture, make sure that UV Set is still selected, go to the Materials tab of the tool bar and activate the UVImage tool. Set a suitable resolution to X and Y fields, for example 400*400 pixels. Change Type to Evaluate and change the file name to a more suitable form, such as 'facemap'. Hit OK and wait until the evaluation is finished.

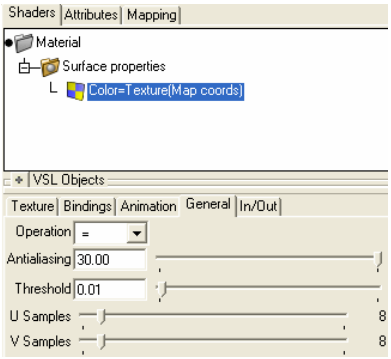


The tool adds a new material mapping under the UV Set object: Default Mapping(SDS UV Set Colormap)



In the material library, you can find the respective material, which textures the UV set area. Open the UV editor; you can see the created texture map as a background image. It may look grainy because of a low default resolution, but you can increase the quality by selecting the UV Set and by increasing the Texture quality value in the Wire tab of the property window. Then UV editor will display the background image accurately.

A test rendering shows that the edges of the new UV Set texture match the surrounding area quite well. However, the new texture mapping material does not include the same antialiasing mechanism as the procedural shaders created earlier in this tutorial. We can add antialiasing to the texture map object, too. Switch to the material library of the select window, select the new material 'SDS/UV/Set/Colormap', open the property window and check the advanced option. Select the Texture object from the shader tree and switch to the General tab. Set Antialiasing to a high value, for example 30, and the Threshold to 0.01. U and V samples can be for example 8. The higher the value, the better antialiasing you will get. 8*8 samples out of a 400*400 pixel image is a reasonable compromise between speed and quality.

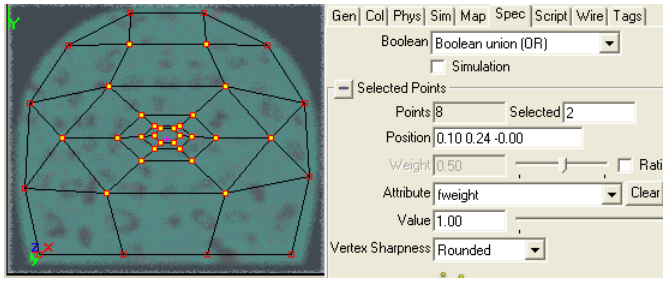


You can test antialiasing factors by making a couple of test rendering cycles. Skin patterns should appear sharp when you view the object from a close distance and fade away when you move camera further away.

Both the procedural skin and the painted skin have now antialiasing enabled. This does not ensure continuity at the border, because mechanisms are quite different. Therefore, we will next set up a gradual fading of the texture map towards its boundary. We need a new channel for that purpose. Switch to the Channels tab of the select window and create a new channel using the popup menu New/Float Channel. Rename the channel as 'fweight' or something similarly descriptive - the channel is the weight for the face texture map.

Switch back to the geometry tab of the select window. Open both the UV editor and the property window. Go to the Spec tab of the property window. UV Set should be still selected and in the edit mode. If not, you can select 'Edit Points' from the UV Editor's popup menu. Use for example the lasso selector (hotkey '4' + LMB drag) to select only the interior points of the UV Set.

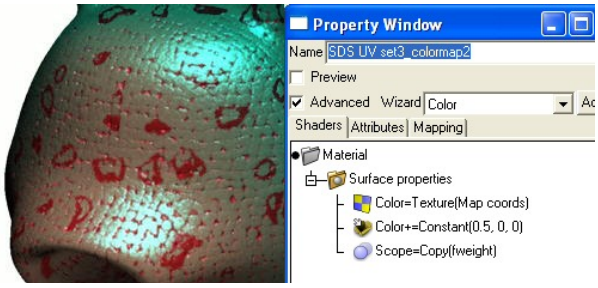
On the property window, choose fweight to the Selected Points/Attribute field. UV Set has no values assigned to this channel, so the value field is empty. Type the value 1.0 and hit enter. Now the interior has full weight, but vertices at the UV Set boundary got a default channel value 0. You can verify this by activating one of the boundary points.



Close the UV editor. The UV Set is ready, so you can turn off its wireframe using the WF-Invisible button in the toolbar or the 'i' hotkey.

Switch to the material library of the select window. The last material, SDS UV Set Colormap, should be still active and in the Advanced edit state. To enhance visibility of the blending process, add a Constant object after the Texture object, set its value to e.g. medium red (0.5 0 0) and change General/Operation to +. In other words, we make the face area redder as other parts of the body. A test render shows now a very clear boundary between the materials.

In the VSL editor, drop a Copy object to the end of the shader. Set Source to Surface:fweight and Destination to Surface:Scope. The blending setup is now ready - test render to see that materials blend smoothly.



The Scope channel is the system channel, which defines how strongly a material affects the target. Scope 0 makes the material invisible and Scope 1 gives a full effect. In the configuration we created above, Scope values were taken from a vertex channel, which gives us is a simple and intuitive control.

Advanced users might wonder why we did not assign the scope channel directly to the UV Set vertices but used another fweight channel to carry the value. The reason is simply the order of material evaluation: the Default Map under the UV Set is evaluated last. Its default scope 1 would overwrite the scope values from UV Set vertices. Therefore we use another, private channel to carry the values to the required shading level.

Now you can finish the face texturing. Paint some details to the created texture map (realsoft3d/facemap in the example above) in your favorite paint software, or use the 3D painting tools of Realsoft 3D. Then save the project.

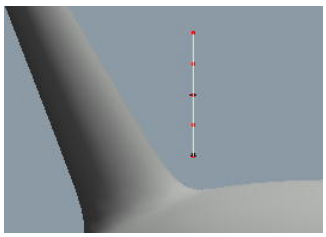
Hair

Next we will add some hair to Harry's head. You can continue the tutorial from the previous section by loading the sample file 'tutorprojects/harry/body_and_uvskin.r3d'.

We start hair modeling by drawing a single hair element: a NURBS curve.

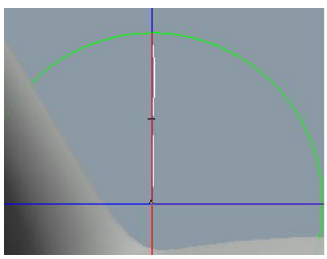
So, go to the NURBS tab of the toolbar. Activate the Curve tool and draw a single hair into the view window.

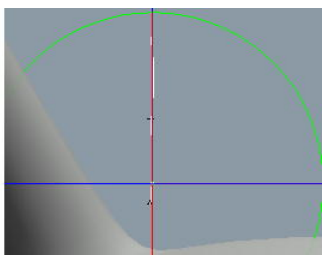
If you want to create long hair, use a sufficient amount of points so that hair bending, curly hair etc. details can be represented by the geometry. In this example, we will make relatively short hair, so, 5 or 6 points are quite enough. You can place the points at regular intervals. Position of the curve does not matter. Its shape should be quite a straight line - hairs will be bent and made curly later. After the last point, select Accept or hit the Enter key to finish the curve tool. Then rename the curve as Hair.



Each hair will be positioned to the skin using the pivot point - the center point of red, green and blue object handles. To ensure that hair will properly stick into the skin surface, not float above it, we have to move the pivot point along the hair.

You can do this by carefully clicking the center point of the object handles, and then a second time, say, 10% towards the opposite end of the hair. The pivot point can be moved also by activating the move tool ('m' hotkey) and selecting Target=Handles from the tool control bar.

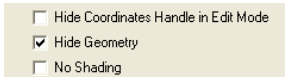






Note

The hair will be added using the Paste tool. The tool rotates the pasted objects so that their blue object space axis matches the surface normal. Therefore, make sure that the hair object you created has a correct object space orientation. The points should approximately follow the blue move handle line. A new NURBS curve has the correct orientation after its creation, but the orientation can change if you point edit the curve heavily or apply tools, which change the orientation, such as the Normalize Object Space tool.

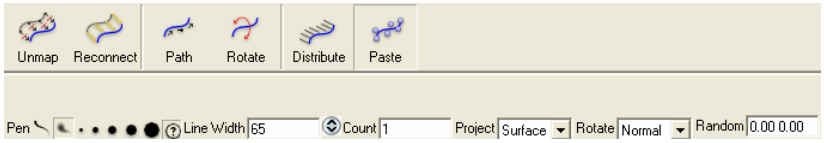


To ensure quick drawing of a large number of hair objects, open the property window and go to the Wire tab. Check the Hide Geometry option and set Show Control Polygon=Always. Control polygon is the simplest way to draw the curve geometry. It is not totally accurate but good enough for our purposes. Clear the Draw Knots option.

While the curve is still selected, switch to the Spec tab of the property window. Set both Head and Tail radius to a suitable value - you may try 1 mm first. It's quite thick, but we can adjust it later. Now the hair will have a constant radius, just like human hair. Using a lower tail radius to get a tapered shape is naturally possible. Because of antialiasing related reasons, infinitely thin zero tail radius is not recommended for animated hair. It is wiser to use 'almost zero' radius such as 10 % of the Head radius value. Decrease the Quality from the default value 24 (suitable for scanline) to the minimal value 2. That is most likely sufficient and renders much quicker. Then switch to the leftmost Gen tab and clear the option Invisible in photo realistic rendering.

Next we scatter some hundreds of hair copies on top of Harry's head. We do not have to add every single hair, but only a number, which is enough to define the hairy area, hair growth direction and other such variations. The rest will be generated by interpolating the sample hairs.

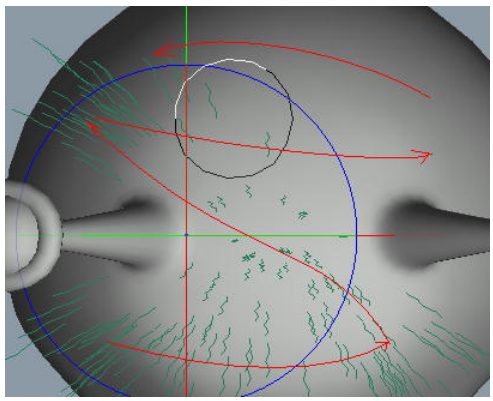
Make sure that the curve object - the sample hair - is still selected. Then take a top view over the head. Go to the Lattice mapping tab and activate the Paste tool. Select the Airbrush option, set Count = 1, Project=Surface and Rotate=Normal.



Move the mouse pointer over the view window. The brush size is shown as a circle. Adjust the diameter to approximately one quarter of the diameter of Harry's head (with the scroll wheel of the mouse or from the tool bar). Too large radius will scatter copies too distantly, making interpolation hard. Too small radius just means lot of pasting work to cover the head top.

Press LMB down to start pasting and move the mouse around at a constant speed to add hair. For an optimal result, try to fill the surface by pasting hair bands back and forth across the head. Make sure that these hair bands overlap enough; otherwise you may see visible gaps between them.

While pasting, you can release the LMB, change the view direction and continue again. If possible, continue new pasting from where you ended the previous brush strokes. You can also adjust the brush radius any time. Try to avoid brushing hair to the horns and other undesired places. However, misplaced copies do not matter much because we can relatively easily delete them later.



When a proper area of the head is covered by hair samples, click Accept to finish the tool. If you are not happy with the result, apply Undo and hit 'a' hotkey to start pasting again with the same options.

When hair pasting is done, the select window lists a large number of new objects. Curve duplicates are automatically selected and you can see their count in the top area of the select window. For a simple character

like Harry, 500-1000 hair samples can be enough. A high number of hairs will unnecessarily slow down further editing and especially animation actions. Therefore, it is wise to aim at a minimal sample count.

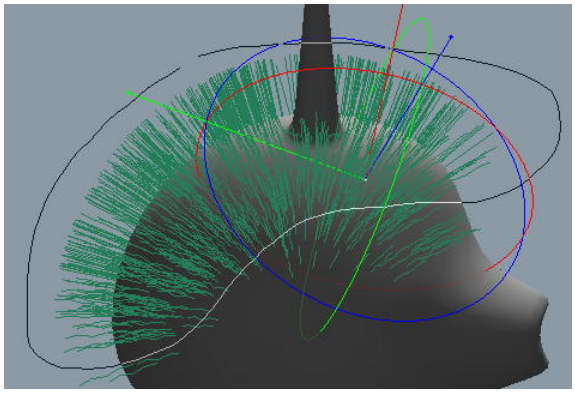
While the new hair copies are still active, select Drop to a level from the popup menu of the select window. Delete the original hair sample. Drag and drop the level containing the hairs from the root level into the level called 'harry' - hairs belong to the hierarchy of the character.

Open the hierarchy level that contains the hair curves on the select window (just click the little box at the left side of the level icon). After opening the level hierarchy, you can select individual hair objects again.

You can trim a symmetric hairline, and the same time, remove unwanted hairs as follows. Select any hair on the select window and hit Ctrl+a (or apply the pull down menu Edit/Select All). Then take a side view, zoom and pan so that you see the hairy area.

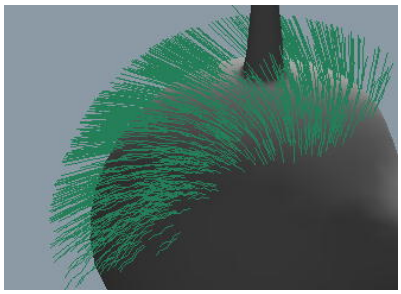
Press Ctrl key down to activate AND selection. Then press also '4' key down to activate lasso selection. Both keys held down, draw a loop that encloses the hairs you want to keep. Only the hairs inside the lasso remain selected.

Select Edit/Invert selection from the pull down menu (or use Ctrl+i) and hit the Del key. Undesired hairs are removed. If there are still some unwanted hairs left, for example around the horn and skull joining area, just zoom in heavily and click the hairs one by one and delete them by pressing the Del key.



Trimming the hairy area with the lasso selector

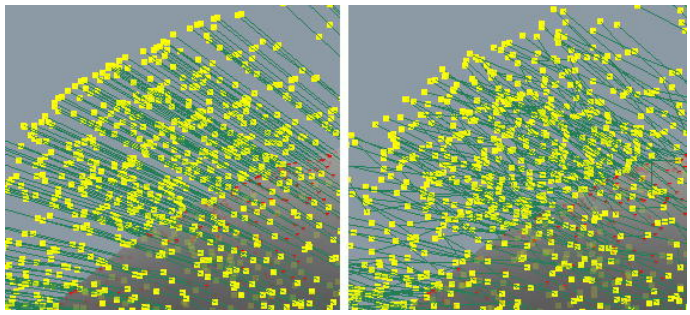
Let's experiment how to adjust hair direction and shape. Now all hairs are pointing perpendicularly away from the head. Select again all hairs like we just did above (use Ctrl+a). Take a side view, activate the rotate tool ('r' hotkey), set the About Pivot option from the toolbar, click in the middle of the head and then somewhere at the outer edge of hair. Move the mouse and observe how each hair rotates around its root. Rotate hairs so that they turn backwards and Harry gets a bit more aerodynamic haircut.



Each sample hair still has an identical shape, which is unnatural. To randomize the curves, make sure that all hair objects are still selected. Hit spacebar to put them into the point edit mode. Open the property window, go to the Spec tab, activate the Selected point field, type in 0 and hit Enter key. This selects the root point of every hair. Actually we do not want to dislocate the root point from the skin - it is the only point that must not move. So, apply the pull down menu Edit/Invert selection or press Ctrl+i. Now we have selected a suitable point set for geometry randomization.

Randomization can be applied using the Noise transform tool. Zoom in heavily to see part of hairs in detail. Go to the Transformation tab of the toolbar and activate the Noise tool. Click once in the view window; position does not matter because noise fills the whole space.

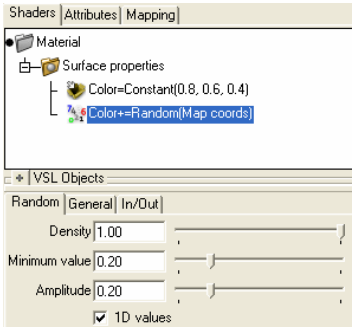
The second click position, however, is important. If you click close to the first point, the noise field will be dense and every single point moves quite randomly. If you leave more distance between the clicks - say a length of a hair or more - noise deforms hairs into recognizable turbulence patterns. After the second click, moving the mouse away from the click position increases deformation. The third click finishes the tool. Experiment freely with various distances - just hit Esc to cancel the tool and try again, or use Undo afterwards.



Randomizing hair points with the Noise transformation

So far, we have not defined any color for hair. We will now assign a unique color to each hair using a material. Hit spacebar to turn off the edit mode and select the parent level of the hairs. Go to the second (materials) tab of the select window. Select New/VSL material from the popup menu. A new material is added at the bottom of the list. Open the property window. Rename the new material as 'Haircolor' using the Name field. Select the 'Color' wizard and hit the Add button. A new Color field appears to the property window. Click the color square and pick a base color for the hair from the palette, for example a light brown (R=0.8 G=0.6 B=0.4). Then check the Advanced option. Click Surface properties in the shader tree to select it, open the popup menu and select the item New/Random. Set Density value of the created Random VSL object to 1.0 (to randomize everywhere), Minimum value to 0 and Amplitude to 0.2. Check also the 1D option if you just wish to change the darkness of hairs. 3D randomization will make hairs slightly colorful. Finally, in the General tab, change Operation from '=' to '+'. In other words, the material will make each hair a random amount lighter.

The material is now ready and you can close the property window. Make sure that the new Haircolor material is still selected. Select Map/Parallel from the select window and draw a parallel map box into the view window with 2 clicks. Map size and position do not matter, because the material simply picks a random value in each point.





Note

By using a Noise object instead of a Random object, you can assign hair colors that change less rapidly. With a suitable material, you can define color stripes or other patterns.

Switch back to the leftmost tab of the select window. Select the first hair object and then Edit/Select All to activate every hair object and the newly inserted parallel mapping, which is the last object in the level.



Note

If Select All for some reason does not highlight all hair curve names, then you have left the point edit mode on, and the first curve has all points selected. If this happens, scroll down to the end of object name list and Ctrl-click the last name to range select all objects.

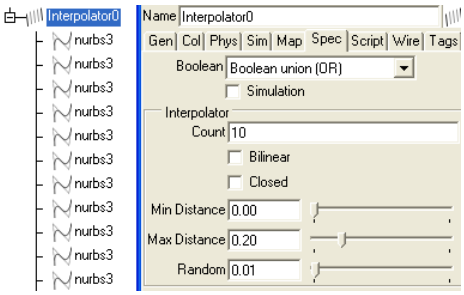
Go to the Materials tab of the toolbar. Activate the Map2Obj tool. Set Target=Object, Source=Color and Destination=Color. These settings will copy material defined color into object colors. Then hit Accept. Now you can find, select and delete the parallel map object from the select window. We do not need it any more because colors are now stored to hair objects.

Time to save the project! Then find a suitable view angle to hair and hit the zero hotkey '0' to render the view window. The hair looks OK but is far too sparse. We will now add the already mentioned interpolation system, which produces more copies. Once again, select all hair curves. Go to the Creation tab of the toolbar, activate the Interpolator tool and click Accept.

With the new interpolator object selected, open the property window and go to the Spec tab. The default interpolation count 10 is quite OK - it will multiply hair count by 10. Set Min distance to a small value such as 0.001. That ensures there will not be 10 overlapping hairs within one-millimeter distance. Set max distance to a

value, which is about 10%-20% of the diameter of the head. A value 0.1 prevents interpolation across distances higher than 0.1 meters. The idea is to interpolate curves that are 'relatively' close to each other. Interpolating a curve on the back of the head to another one on the forehead is useless because interpolated copies would be placed inside the skull. Another example: you would not want to interpolate cat's tail hair towards a hair in its ear because interpolated copies would float in the air.

We can also add some geometry randomization here: set the random value for example to 0.01. This makes each interpolated copy unique. Do not set too high random value because hair can get dislocated from the skin. Do you remember when we positioned the pivot point of the first hair? The random value should not exceed the distance of the pivot and the first point of the curve.



Now interpolator is ready. Render a test image ('0' key) and adjust its parameters if necessary. Bilinear interpolation often gives a better result, especially if the number of used samples is low.

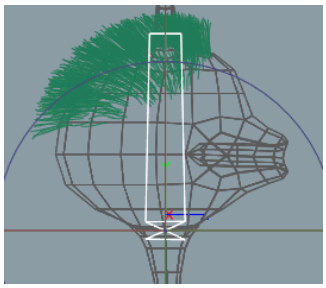
Save the project with a new file name, such as 'hairy_harry', to backup your work.

Skeleton

The next task is to bind Harry to a skeleton. The example project for starting this chapter is 'tutorprojects/harry/body_hair_skin.r3d'.

Take a side view. Set view projection to Parallel (hotkey '*' toggles the projection), because it is easier to position the skeleton without the perspective distortion. Zoom the view so that the body fills it. Placing skeletons inside shaded surfaces is often easier in wireframe mode. If your view window is still in Shaded OpenGL mode, select View Properties from the popup menu, go to Draw tab, clear the Shaded option and close the View Property Window.

Switch the tool bar to the Modifier tab. Activate the Skeleton tool. We will draw the backbone first. Because the body shape is so simple, one bone will be enough to control the body. So, LMB click first in the hip, move the mouse upwards. Swing the mouse sideways (approximately 45 degrees to both directions) to define a typical motion range of the backbone. At the start of the backbone, there is a triangular joint constraint, which shows how much the bone can rotate around its root. The triangle should point upwards. If not, move the pointer straight up from the hip and Ctrl-Click to set the constraint direction. Click second time at the top of the head to define the end point of the backbone. Then click Accept or hit Enter to finish the tool. Rename the created object as backbone.

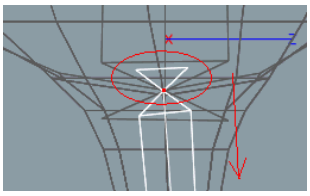

Backbone added

Next we will draw a skeleton for legs. Motions of bones are constrained by a set of angles. You can define the most important angle constraints intuitively already when drawing the skeleton. Therefore, it is best to draw the skeleton in a plane where it bends most. Clearly, a side view is the best view for defining how the knee bends.

Activate the skeleton tool and click first in the hip. Leg will connect there to the backbone. Then move the mouse down towards the knee. Try to avoid accidental mouse movements upwards from the hip because mouse motions now define the main angle constraint for the joint. Harry is a fat character and his legs cannot bend that high.

We can set now the orientation for the hip constraints of the leg. The constraint space should point towards the center of the motion range of the thighbone. For a leg, that direction is straight down. This is quite important: if the constraint space points upwards, key framing leg's motion from back to front would rotate the leg up through the body.

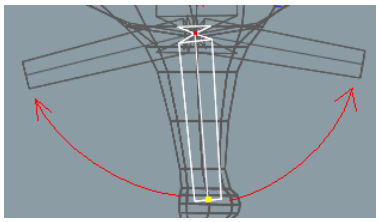
You can see the direction of the first constraint from the triangle at the start of the skeleton. If you moved the mouse down from the hip, the triangle can already point to the right direction. If not, press Ctrl down and LMB click in the middle of the knee. Ctrl-click does not add new joint but sets the constraint orientation.



Ctrl click at the knee sets root constraint direction downwards

Move the mouse forward and up until the thighbone is rotated to its extreme front position. Then rotate the bone back to the knee position and then backwards to its extreme back position. This way you can 'teach' to the bone how much it can move.

Click in the middle of the knee to add a new joint. Then carefully move the mouse down to the angle point. Define the angle limits again by moving the mouse backwards and up, until the shinbone is rotated to its extreme position. Do not rotate the bone forward, because the knee does not bend that way. Click in the ankle to finish the shinbone.



Defining angle limits for a bone

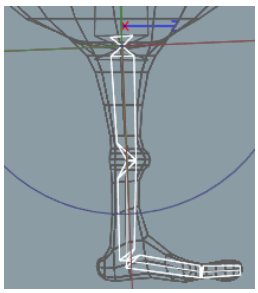


Note

If you accidentally define a wrong joint position, just select Go Back from the popup menu or hit Backspace. You do not have to start from the beginning. Note also that angle limits can be adjusted later easily. It is enough to draw the skeleton approximately the right way. You can fix and fine tune everything later.

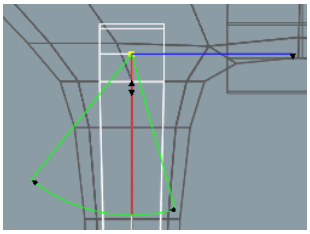
After the ankle joint, add one joint to the position where toes bend. Put the last joint to the tip of the toes. Click Accept to finish the leg skeleton. Rename the skeleton object as leg1.

Switch to the front view. Make sure that the backbone is in the middle of the body. Move it, if necessary. Select the leg skeleton. Use the blue object space handle to move the skeleton to the middle of the left leg.



The skeleton for legs finished

Hit the spacebar key to set the skeleton in edit mode. Click the little red knob at the hip joint to select the first joint of the leg. You will see some handles, which control constraint angles. Drag the little black triangle at the end of the green line to the left, until the angle sector appears suitable for thighbone's motion range to that direction. Then drag the triangle at the opposite end of the sector to the right.



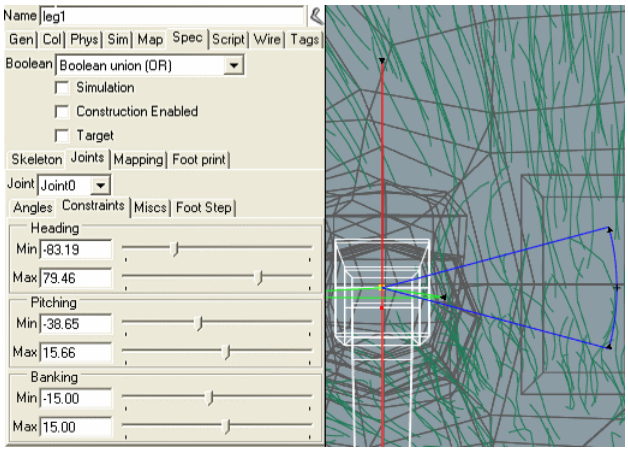
Adjusting angle constraints for leg motion sideways



Note

The cursor changes into a pen shape when the mouse pointer is close enough for dragging a constraint handle.

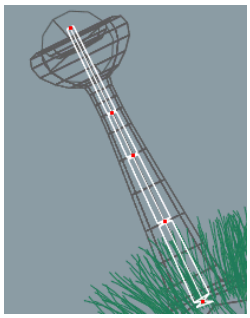
There is a third motion sector to define left: rotation of the thighbone around its axis. Take a top view. You can see the sector as a blue line. It can be adjusted just like the green sector above, but let's use numeric controls for a change. Open the property window, go to the Spec tab, and Joints/Constraints sub tab. Decrease the Banking/Min angle to, say, -15 degrees, and Max angle to 15 degrees. Note how the sector changes respectively on the view window. Close the property window.



Adjusting banking constraints from the property window

Now the first leg skeleton is ready. Hit spacebar to turn the edit mode off. Switch back to the front view. Press Ctrl down and move a duplicate of the leg skeleton using the blue move handle to the middle of the other leg. Rename the copy as 'leg2'. Turn the edit mode on and adjust the green motion sector of the thighbone (it should be mirrored from the other leg).

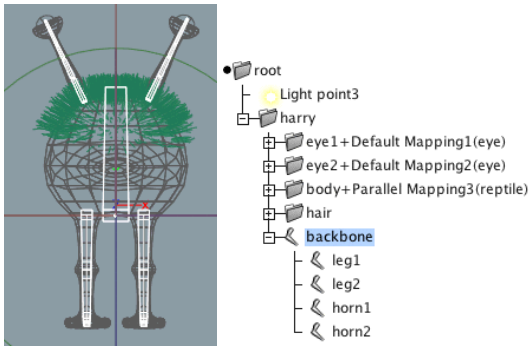
Legs are now ready. We need to add two skeletons more for the horns. We can assume that horns are quite flexible and bend to all directions symmetrically. So, we can draw the skeleton from the front view. Pan and zoom the view to see the left horn properly. Start the skeleton tool and draw the skeleton inside the horn the same way as you defined the leg skeleton. Remember to set the root constraint after the first point with a Ctrl-click, and swing the mouse slightly sideways to set symmetric angle constraints for each joint. For flexible bending, add as many skeleton joints as there are cross sections in the SDS control cage. When the bone and geometry density match, the result behaves elastically. You can end the skeleton anywhere after the bendable area. It does not have to reach longer than controlled geometry points. Rename the new skeleton as horn1.



The skeleton for the left horn

We need to allow some bending of horns in front-back direction as well. Set the horn1 skeleton to the edit mode. Click the first joint (the one close to the head) to select it. Open the property window and go to the Spec/Joints/Constraints tab. Enter the Pitching range Min=-30, Max=30 degrees. You can finetune the Heading limits to the same range if you like. Banking range can remain zero because horn shape does not need it. Activate the joints one by one and set same kind of angle constraints for them all. Then you can close the property window.

Duplicate the horn1 skeleton, move and rotate it inside the second horn geometry. Rename the copy as horn2. Take a side view and make sure that the horn skeletons are inside the respective geometry.



Multi select the leg and horn skeletons and drag them into the backbone hierarchy. Multi select all the sub skeletons, and hit the Attach button on the toolbar to connect the sub skeletons to the backbone. The Attach option means that pulling any bone with Inversed Kinematic tool will eventually drag the full skeleton along (unless so called anchors have been set). Select the backbone and click the Store button on the toolbar. This records the current state of the skeleton (including the sub skeletons) to be the native pose of the system. You can any time return to the stored pose by clicking the Native button. Finally, drag and drop the backbone object into the hierarchy of the Harry character.

The next step is to bind the points of the model to the skeleton. The project template for practicing the binding tools is 'tutorprojects/harry/skeleton.r3d'.

Select the backbone and hit the Bind Nrst (Bind Nearest) button in the toolbar. This operation, which binds all objects in the same hierarchy level, is a simple way to get started, although the result most likely needs some corrections.

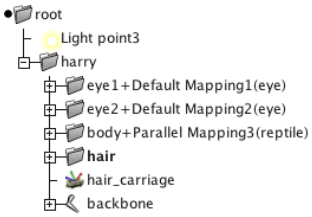
Make sure backbone is still selected and hit the spacebar to put the skeleton system into the edit mode. To test the result of the Bind Nearest action, drag the skeletons for example from the end of a toe bone to change the pose. Most likely you will see some strange deformations in the geometry of the character, because of some wrong skeleton bindings. Hit the Native pose button in the toolbar to reset the skeleton.

Apparently eyes and horns were mapped correctly - the respective skeletons are clearly the closest. Hair should be bound only to the backbone, but part of it got bound to horns. The simplest way to fix this is to unbind the hair completely and then make a new binding. So, multi select the hair level and the skeleton system's root object (=backbone) and hit the Unbind button in the tool control bar.

Hair no longer follows the skeleton. We can also clean up the model at this point. Open the hair level, then the interpolator object under it and select the first hair curve. The bold typeface on the select window indicates that the hair curves still have animation data attached (the root choreographs, under which the skeleton binding choreographs were placed). Make sure that the first curve is not in point edit mode and hit Ctrl-a to multi select all hair curves. Then select Delete Choreographs from the select window's popup menu.

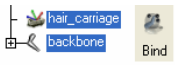
The skeleton we created will not be used to deform the hair - it is far too inaccurate for that purpose. Thereore, we can use a simpler and quicker way to attach the hair to the backbone. Instead of mapping the thousands of points one by one to the backbone, we will use a special Carriage constructor object to attach the hair object as a single target.

Close the hierarchy of the hair level and leave the level selected. Go to the Modifier tab of the toolbar and activate the Carriage tool. Click in the middle of the backbone, move the mouse and click a second time. The two clicks define a coordinate system, which will control the hair. The orientation of the coordinate system does not matter, because all its points will be attached to the backbone. When the tool is finished, a new item 'Carriage' appears to the select window. Drag and drop it to the hierarchy of the harry level, and rename it as 'hair_carriage' to describe its purpose better.



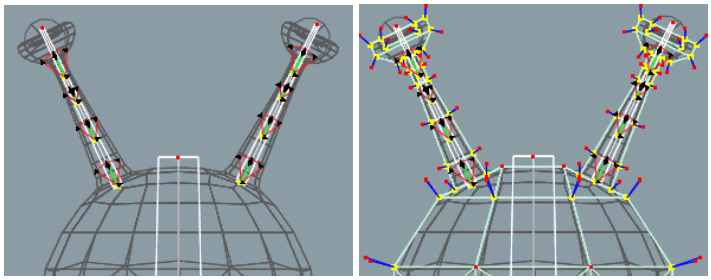
Put the new carriage object into the edit mode (hit spacebar) and select all its points by dragging a box around the coordinate system geometry. Next select the backbone object. It should still be in the edit mode. In the view window, click the bone inside Harry's head (= the real backbone) to select it. Remember: the selection is indicated by a highlighted bone axis and the constraint handles.

Then shift click the carriage object on the select window, to multi select both objects. Click the Bind button of the skeleton tool group on the tool control bar (note: use the Skeleton Bind, not the Carriage Bind tool). The geometry of the Carriage object became bound to the backbone.



The next goal is to fix the incorrect bindings of the body mesh. Multi select the hair level and the carriage object and hit 'i' hotkey to make the dense distracting wireframe invisible. Select the body level and hit spacebar to turn the edit mode on. If you do not see the point handles, open the level, select the actual SDS body under it and change its edit mode to Edit/Points from the tool control bar. Then close the level containing the body and leave it selected, edit state on. Make sure that all points are unselected - hit Ctrl + t to unselect all points, if necessary.

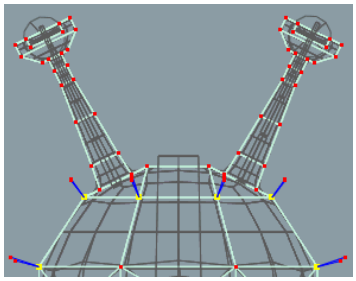
Select the backbone skeleton object. Drag select all bones on the left horn. Then press Shift key down and add all bones of the right horn into the selection by a new drag selection box. Shift click the Body level on the select window to multi-select it, too. Then hit the Select button on the tool control bar. All points mapped to the horns became selected.



Applying the Select tool to find out which points were mapped to the horn skeletons

Click the body level on the select window to remove the skeleton system from the selection. Then deselect all points of the horns of the body, for example by Alt-dragging two selection boxes around them. The remaining selected points are exactly the points, which were incorrectly mapped to horn bones instead of the backbone.

Press Shift key down and add click the Backbone object on the select window to multi select it with the body. The skeleton still has the horn bones selected after the previous steps of this tutorial. Click the Unbind tool to remove the incorrect binding.



Incorrectly mapped points extracted

Carefully change the object selection to the skeleton only by clicking it on the select window. You should not lose the valuable point selection of the body mesh by an accidental click or drag on the view! When the skeleton system alone is selected, go to teh view window and click the backbone inside the body to select it. Then Shift click the body level on the select window to multi select it. Finally, hit the Bind button on the tool control bar to make a new, right kind of binding.

There are some more incorrect bindings at the leg-backbone joint area, but you can fix them exactly the same way as instructed above. Just select all leg bones, apply the Select tool to find out which points are mapped to them, exclude the correctly mapped points, unbind the remaining points from the leg bones and bind them to the backbone.

Save the project as 'harry_rigged.r3d'.

Animating the Character

In this section, we will add facial expressions and skeleton movements to the project.

The example project for practicing animation steps: 'tutorprojects/harry/rigged.r3d'.

To see Harry's expressions better, turn shaded OpenGL on. Then, select the SDS object, which represents the body and set it to the point edit mode. If the animation window is not opened yet at the bottom of the screen, open it from the View pull down menu or by hitting the hotkey F7. The number of frames, displayed on the animation window, can be for example 100. We can change it later if necessary.

Open the choreography window (Animation/Choreographs from the pull down menu or F6 hotkey). Go to the Time Lines tab. There is a list of choreographs on the left side of the window. Open the popup menu of the choreography list gadget and make sure 'Only Selected Objects' is enabled. Otherwise the hundreds of skeleton mapped hair choreographs may show up although they will not be edited.

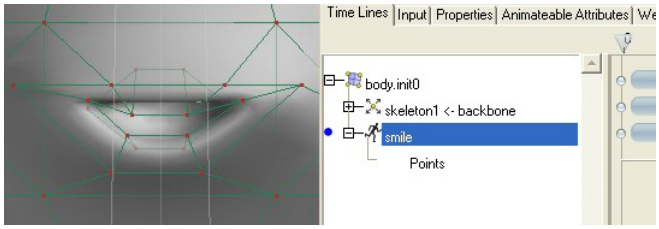
Now the choreography list shows two items: 'Body:Init' and 'SDS UV Set:Init'. Click the latter one and select Delete from the popup menu. We will not animate the UV Set, so we can clean up the list this way.

You can open the contents of the Body's root choreography by clicking the little square at the left side of its name. It includes one item 'Skeleton1<-Backbone'. This is the choreography, which binds the body to the skeleton.

We will define a couple of expressions using a simple pose approach. Select the Body:Init choreography by clicking it and then New/Pose from the popup menu. A new item appears to the end of the choreography list.

If you open its sub structure, you can see that it contains one animated attribute, Points. The attribute was automatically included in the pose because the previous skeleton chorography modifies geometry points.

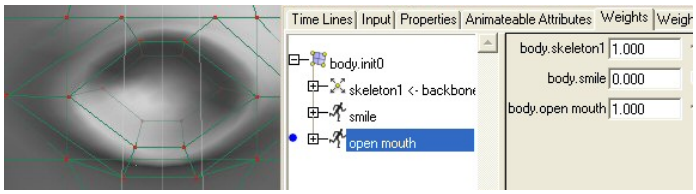
Points are just what we want to pose, so we do not have to make any changes in the attribute list. However, Select the pose item and apply Rename from the popup menu to rename it as 'smile'. In the view window, move the points of the mouth to form a smile expression.



The first pose: Harry is smiling

Go to the Weights tab of the choreography window. The weight of the new pose is at the full level 1.0. Move the body.Smile slider down and up, and observe how the expression changes respectively. Leave the slider to zero position. This makes it easier to define more expressions. Poses are combined by adding the weighted point movements together, and it is best to define each pose starting from a neutral expression.

Select New/Pose again (you can have the existing pose or the body:init0 choreography selected when doing this). Rename the new pose as 'Open Mouth'. Then edit the mouth area to reflect this new expression. Try the weight slider of the new pose to open and close the mouth. Leave the weight to zero.



The second pose: the mouth wide open

You can add as many expressions as you like this way. Let's see how we can use the poses in animation. But before that, there is one important change we must make: the order of choreographs is wrong. Namely, the skeleton choreography should be after the face expression poses in the choreography list.

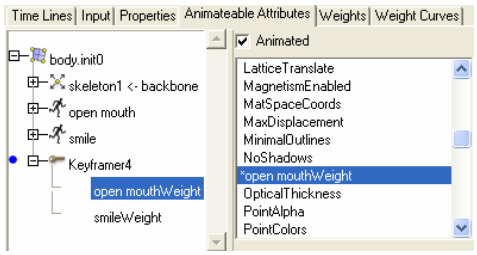
For example, the smile pose moves mouth sides upwards (up direction being defined in the object space of the body). If the skeleton had bent Harry's face to look down, movement upwards would move the points of the mouth inside the characters head, which is apparently wrong. To change the order, drag and drop the skeleton choreography after the two poses.



The skeleton deformation comes after the poses

Make sure that the body.init0 choreography is selected and pick New/Keyframer from the choreography list's popup menu. Click the new Keyframer item to activate it, and go to the Animateable Attributes tab. Find the attribute SmileWeight, select it and set Animated on. Do the same for the Open Mouth Weight.

The keyframer also includes Points as an animated attribute (it was included because the previous choreographs had it, too) - you may clear the animated state of that attribute. To do it, select Points from the attribute list and uncheck the Animated field. The keyframer will now only control the selected weights.



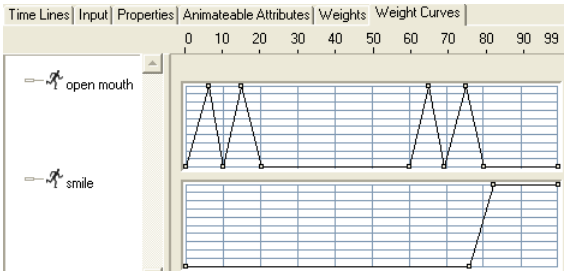
The pose weights prepared for key framing

Switch to the Weight Curves tab. Select Create predefined curve/45 degree curve from the popup menu of the curve gadget at the right side of the smile item. Change the curve type to Poly line - it is continuous enough for controlling expressions, and very easy to set up. Then edit the curve so that Harry starts smiling at the end of the animation. To achieve this, click the curve about at the frame 75 and move the inserted point down to zero level. Then click the curve about 10 frames later and move the new point up to the full level one.

Note: The Weight Curves tab shows the curves, which control choreographs. The program does not draw any numerical scales to keep space requirements minimal. This is important, because often several curves must be visible simultaneously for animation synchronization. The vertical range of the curve graphs is by default 0..1, which is suitable for weight control. You can examine and edit a single weight curve more accurately in

the Properties tab, or adjust the graph to include some negative weight range, too. A smile with a weight -1 is an unhappy expression.

Apply the same two menus, Create predefined Curve/45 degree curve and Type/Poly line, to initialize a curve, which controls the Open Mouth choreography. This time edit a curve, which opens and closes the mouth a couple of times in the beginning, then again during the third quarter of the time line. This Curve makes Harry talk. Make sure that the weight goes to zero when smiling starts. Otherwise the expression - a smile with a fully opened mouth - might look strange.

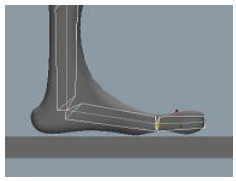


Weight curves, which make Harry talk and smile

Now you can play the animation and check that everything works as expected. The playback speed may be somewhat slow, mainly because of the large number of hair objects connected to the skeleton. You can also open a smaller view window from the Windows pull down menu and apply Animation/Preview from its popup menu to create a quickly playable .avi preview.

Next we add some skeleton controlled motions to the character. For example, let's make Harry jump. Rewind the animation to the beginning. To make animation easier, take a top view and use toolbar's Analytic/Cube tool to create a floor object. Take a side view and move the floor under the feet. Select the skeleton and make sure the skeleton is in its native pose - apply the Native button of the toolbar if necessary.

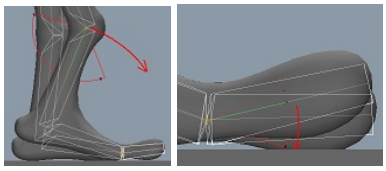
The first phase of a jump is that Harry squats down to get some speed for his jump. This probably takes half a second. So, RMB click on the animation window about at the frame 12. The time slider jumps to that position. Put the skeleton into the edit mode. From the side view, zoom into the feet. Drag-select both toe bones. Set the anchor selector in the tool control bar from Free to Anchor. A little anchor symbol appears to the locked joints. Be careful to lock the joints where the toes start from the feet, not the tips of the toes.



Toes anchored to floor

Turn animation recording on. Move the mouse over the knee position and drag the point handle at the knee forward and down. Dragging activates the inversed kinematics tool, which computes a pose by rotating the connected bones. Do the same for the other knee. Note that you can hit the Esc key to cancel joint dragging at any time.

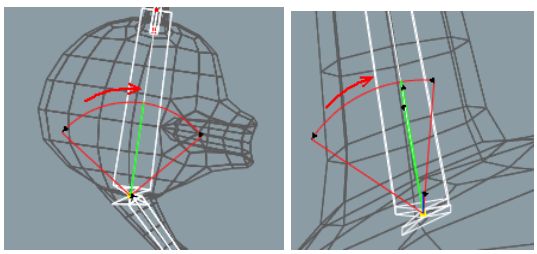
Drag the knees until the legs are properly bent. Then check the toes. It is possible that a toe angle constraint has rotated the toes of a foot when you bent the other leg. If that has happened, click the toe bone and drag from the red arc (forward kinematics handle) to rotate the toes so that they touch the floor.



Drag the knees with IK and fix the toes with FK

Click the backbone to activate it. Drag the red forward kinematics arc to rotate Harry's body to lean forward, so that his position becomes more balanced. Turn Shaded OpenGL off (from the View property window's Draw tab) if it is hard to see the handles inside the shaded skin. Then take a front view and zoom into the root of the horns.

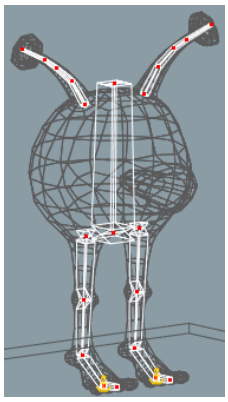
Select the lowest bone of the left horn and rotate it upwards using the FK arc. Be careful and zoom in enough - the bone is quite small, and there are many handles in that area. If you accidentally drag too close of the black triangles, you will start the angle constraint modification instead of FK. If that happens, hit the Esc key. Do a similar rotation for the right horn. The idea is just to make the horns match the motions of the body. These kinds of little details are important in character animation.



Rotate the body forward and the horns straight up from the FK handles

The next keyframe shows Harry leaving the ground. The jump starts with a quick acceleration, so 5 frames is quite enough. Move the time slider to the frame 17. Take a side view. Drag the knees back and up, to straighten the legs. The heels should still stay off the floor. You can also drag the joint at the hip to adjust the body position, and the FK handle of the backbone to make the body point straight up. Because Harry is accelerating the speed upwards, the flexible horns should bend downwards. So, take a front view and drag the horns from their end points to bend downwards.

The next position could define the highest point of the jump. But because the motion up and down is almost symmetric, we will now define the the key position when Harry has just fallen back to the floor, and his feet start resisting the impact of the floor. The highest point of the jump will be defined afterwards.



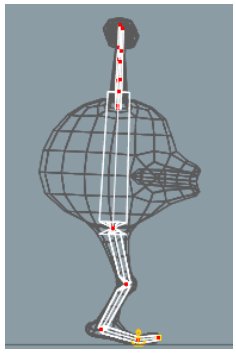
Harry is on his way up

So, move the time slider 10 frames forward, to the frame 27. Now Harry is falling and the horns should point again upwards; apply IK to the horn ends to pose them. From the side view, move the heels slightly downwards.

Now move the time slider back in time to the frame 22. Here Harry has reached the maximum height and just starts falling. Click the backbone to activate it. Press the y key down to lock the movement to the Y axis and drag the spine bone to the up most point of the jump. At this point, the horns have returned to a relaxed position. So, apply IK from the front view to achieve that.

We already defined the key frame (frame 27), where Harry hits the floor. The next key frame shows a squat position, because of the impact of hitting the floor. The squat position can become deepest in the frame 30. Move the time slider to that frame. Apply IK to the knees to squat him, check the toes and rotate the spine bone to lean forward. Don't forget the horns - they bend again down from the impact.

Move to the frame 33. This frame ends the jump. We can easily restore the original position by hitting the native pose button in the tool control bar. Then turn off the animation recording. You can also select the toe bones and release the anchors. The jump cycle is now ready. Rewind and play the animation to see the result.

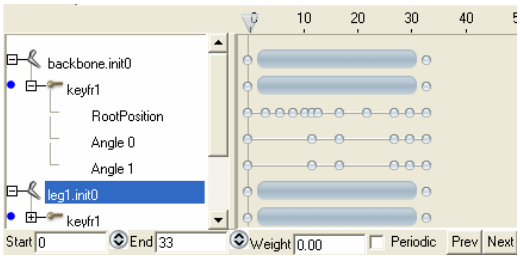


Harry squats when he hits the floor

If you followed the steps above, the motion probably looks quite good. However, some details need adjusting. Most likely the toes penetrate the floor when Harry straightens his legs. Let's fix these errors. Make sure the skeleton system is selected and open the choreography window. Go to the Time Lines tab.

The first thing that might need fixing is the length of time lines. We key framed the jump motion to happen during the frames 0 - 33. The program initialized the time lines to cover the whole animation. It will be easier to position and adjust the timing of the jump if the time lines reflect the key frame range. We will not add any other keyframe motions in this tutorial, and therefore we can set the root level time lines.

Click first the backbone.init0 choreography to select it. Open its sub time lines, including the keyfr1 time line, to see the animated attributes and their key positions. Press the Shift key down click the end knob of each .init choreography, one by one, to multi select it. Do not select any sub time lines! When all .init-related end knobs are selected, press both Shift and Alt keys down and drag the first selected knob to the frame 33. Alt-modifier scales the time line without moving underlying key frames.

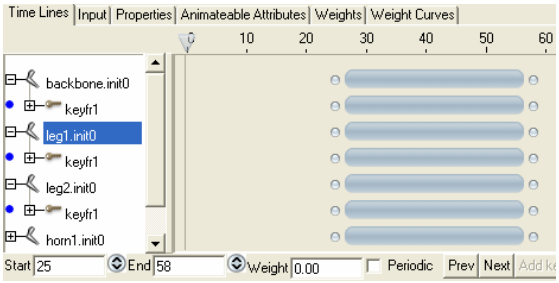


Root level time lines normalized to match the key frame range

The keyframe choreography for the backbone includes some items we will not need. The native pose tool assigned also bone lengths to the skeletons. However, we did not actually change the bone length. A choreography, which assigns a constant value, is pretty harmless, but we can clean up the animation data and improve performance by removing unnecessary items. Click the Length 0 attribute name and hit the Del key

to remove it. Remove also Length 1. Open also other keyframe choreographs and delete the length attributes from them.

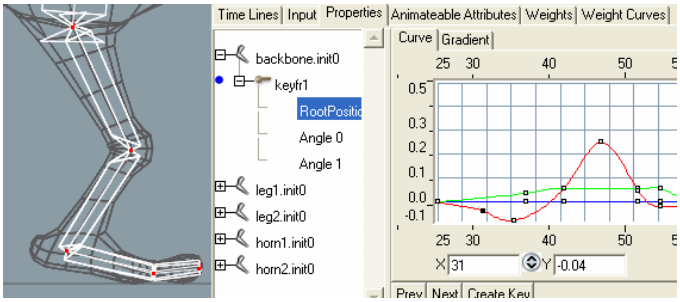
Next we will change the time when the jump happens. Harry is talking in the beginning of the animation - let's make him jump after that, starting from the frame 25. Click in the middle of the time line of the first choreography (backbone.init). Press the Shift key down and select the rest of .init time lines by clicking in the middle of them. Shift key held down, LMB drag the first selected time line to the right, until the start knob is about at the frame 25.



Time lines moved to a later moment

Next we solve the problem of foot penetration into the floor. Move the time to the frame 25, where the jump starts. You can step the animation time back and forth frame by frame either by LMB clicking the time line, using the mouse wheel over teh time line or with the + and - hotkeys. Move time forward, until you find the frame where the feet are in the lowest position. In the example animation, it is the frame 31. Select the choreography attribute backbone.init0/keyfr1/RootPosition and go to the Properties tab. The position curves are defined in the object space of the skeleton. The red curve shows the motion in the vertical direction. Click the red curve in the position, which corresponds the frame 31. A new point appears to the curve. With this new key frame, we can fine tune the motion. Drag the point upwards, until the feet no longer penetrate the floor. The view window is updated instantly to reflect the editing of the animation curve.

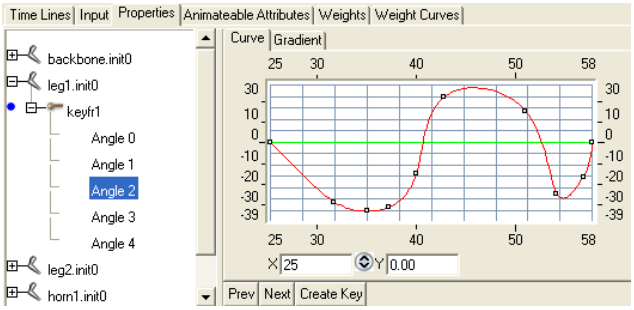
Note that the final height position of the feet is defined as a combination of several bone angles and the base position, which we just edited. Setting y=0 for the new curve knot does not mean that feet touch the floor. It means that Harry's hip is at the same height as in the first frame. If knees are bent or ankles are pointing straight down, feet will not match the ground.



Foot penetration fixed in the frame 31

When you move the time forward, you will notice that some frames later the feet penetrate the floor again. You can add a new point to the red curve and move it upwards to compensate. However, if you solve the penetration by only editing the vertical position of the skeleton, the result may look unnatural.

A better solution is to check all the joint angle curves and edit them as well. For example, leg1.init0/keyfr1/Angle 2 is the angle of the ankle. The ankles do not straighten up in the beginning of the jump. They give the final push for the jump just before the feet get off the ground. Editing the red curve to reflect this fact will create a better-looking motion and reduce the penetration problem.



Ankle rotations improved

Getting the motion right will take a lot of curve editing for all choreographs. It is also important to understand what the curves define, what is the relation of curvature to acceleration etc. Just continue editing the curves and experimenting until you learn to control the motions via the curves.

When you are satisfied with the jump motion, save the project.

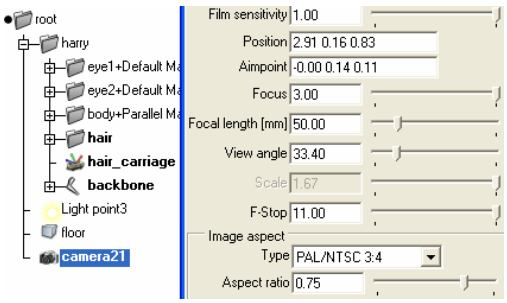
Rendering the Animation

The final stage of the production is rendering the animation to a suitable format.

The example project for practicing this part of the tutorial is 'tutorprojects/harry/animated.r3d'

The animation will be rendered directly to an .avi file. Because the images are not stored from a view window, we must define the camera view for each frame by adding a camera object to the scene. First we define the camera setup for the first frame. Navigate the view window so that you see the animated character for example from the side. Open the view property window, go to the Camera tab and make sure that the settings are OK. For example, you can set the focal length now to 50 mm. Turn also perspective projection on. If the character does not fit into the 50 mm view, just increase the Distance value in the Orientation section of this view property sheet. When the camera setup appears suitable (don't worry too much at this point, you can change every detail later), close the view property window. Go to the Creation tab of the toolbar, activate the Camera tool and select Accept from the view's popup menu. This creates a camera object and records the camera settings of the view window into it.

Double click the camera object in the select window to open the property window. Go to the spec tab. View window most likely has an unusual aspect ratio, which depends on the opened GUI elements, monitor resolution etc. The ratio was stored to the camera object, too. Change the image aspect ratio to some sensible value. In this tutorial, we will use the image aspect of the PAL and NTSC standards. You can select it (height/width = 0.75) from the predefined ratios of the Image Aspect/Type gadget.



Remember to define the aspect ratio

You can adjust other camera settings, too, using the property window. For example, if the image was too dark, you can either make the flashlight color brighter, or increase the film sensitivity value. To preview the result, just drag and drop the camera object from the select window into the view window and render the view. Close the property window.

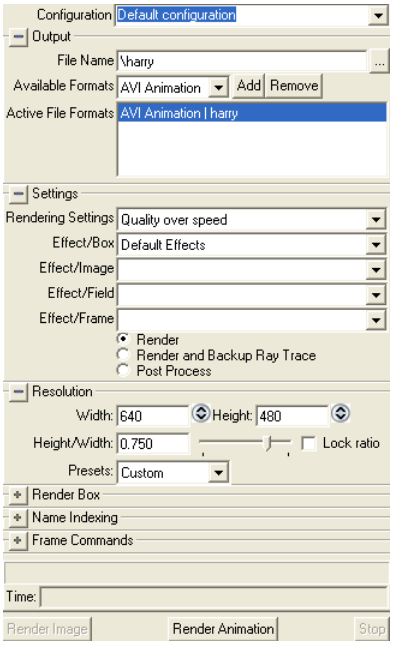
We can animate the camera easily using a couple of key frames. Move the time slider to the end of the animation. Then rotate the view so that you see Harry from the front. Decrease the distance to the character (5th view control from the top in the view control bar) and center the view to show Harry's face. The animation will end to a close-up of a smiling face. Turn animation recording on (use the red button on the animation window). Then select Camera/View to Object from the view popup. This tool records the view camera to the camera object. Because animation recording is on, this added a new key frame for the camera. Turn the animation recording off.

Note: View to Object operation does not assign the image aspect value, because view windows, which are attached to the main window, rarely have a useful image aspect ratio. Therefore, the key frame creation method above did not change the selected aspect ratio (0.75 for PAL) of the camera. If you zoom out slightly, you will see the rectangular wireframe of the camera on the view window. Most likely it does not match the proportions of the view window.

Play the animation to check the camera motion. If the view window does not follow the camera motion, make sure that View property window/Camera tab/Track animation play is enabled.

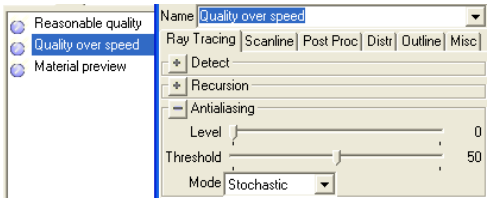
Most likely two key frames are not enough. Stop the playback in the middle of the animation. Navigate the view window so that the main actor is again properly in the picture. Then turn the recording on, apply again the View to object tool and turn recording off. Now the camera motion is ready. Rewind the animation back to beginning.

Select File/Render from the pull down menu. Enter a suitable file name and hit Enter. The default configuration has Windows BMP in the Active Formats list, which defines the outputted file formats. Use the popup menu of the list to remove it. Then select avi from the Available Formats list and hit the Add button. Now the output target is an avi file.



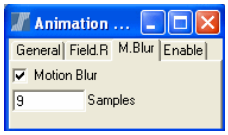
File render window

In the rendering settings field, there is an item Quality over speed selected. It is almost suitable for our project. However, thin hair objects require very careful antialiasing. The best way to antialiase such objects is using sufficient stochastic antialiasing. The important point is that stochastic antialiasing can be obtained as a free side effect of motion blur. On the select window, go to the render settings tab and select the item Quality over speed. Open the property window and go to the leftmost Ray Tracing tab. Set Antialiasing/Level to zero and Mode to Stochastic.



Antialiasing settings for use with motion blur

Close the property window. Select Windows/Animation Settings from the pull down menu. Turn Motion Blur on and set the number of samples to for example 9. Close the Animation Settings window.



Motion blur enabled

Go back to the file render window. We did not define any resolution yet. The official PAL and NTSC resolutions are included in the Presets box. Both PAL and NTSC use non-square pixels, and playing animations rendered to those standards on a computer screen shows the images stretched, because all usual screen resolutions have a pixel aspect ratio, which is quite close to one.

So, let's use a lower, computer-oriented resolution. Enter for example Width=640, Height=480. Note that the aspect field shows a ratio 480/640 = 0.75, which is exactly the aspect ratio we defined for the camera object. This means that the pixels will be exactly square-like (pixel aspect = 1.0).

Save the finished project. Then start rendering by clicking the Render Animation button. Note, that you can safely cancel the rendering, save the project to a temp file, reload it later and continue rendering. When the rendering is done, play the created .avi file with your favorite media player.





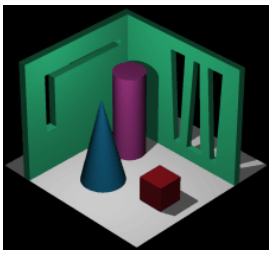
Chapter 3. Modeling

Using Curve, Circle and Rectangle Tools

Realsoft 3D supports three kinds of geometric objects:

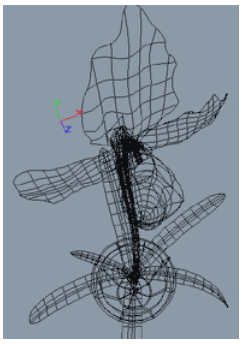
1. Analytic
2. NURBS
3. SDS

Analytic objects are memory efficient, fast to render and they perform fast Boolean operations. Analytic means that the surface and other geometric properties of an object are defined accurately using a mathematical equation. For example, an analytic sphere is defined by two attributes: center and radius. The problem with analytic objects is that due to their minimal definition, their geometry is very rigid and doesn't allow you to represent arbitrary freeform shapes.



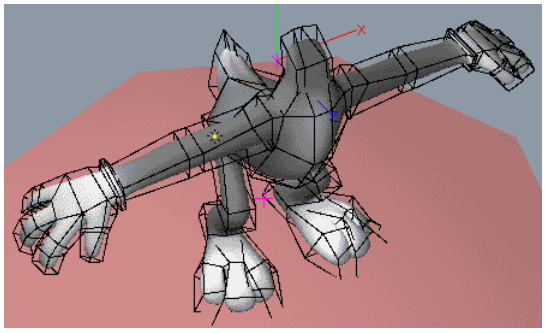
Various Analytical objects - Polyhedron with boolean holes, cylinder, cube and cone

NURBS (Non-Uniform Rational B-Spline) has become a standard curve and surface representation method in computer graphics. NURBS objects are slightly slower in Boolean operations but they can be used for representing freeform shapes easily, because a NURBS surface is defined by a set of freely editable points. One powerful feature of NURBS surfaces is the ability to cut away part of the surface with trim curves. A NURBS surface always has a regular, 'rectangular' topology: It consists of a grid of connected rectangular patches. Branching surfaces can be modeled by joining several surface patches together, and cutting with trim curves creates holes.



NURBS object with trim curves defining edges

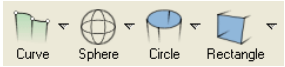
SDS (=Subdivision) surfaces allow arbitrary polygonal and smooth freeform shapes. The major advantage of SDS surfaces is their free topology: A single SDS object can include branches and holes. Therefore, they are very suitable for modeling organic shapes, human figures, etc.



SDS object - created from an SDS cube

The toolbar or Realsoft 3D includes a specialized tab for each of these three object types. You can find basic geometrics shapes from each tab. You can also use so called unified toolset to create geometric objects. The unified toolset can create all geometry types described above, and even more: the four general purpose tools have a large number of sub options. By combining the sub options, you can access hundreds of different tools.

The default toolbar does not include the unified tools, but you can add the tools to your program interface using the Customization/Available Objects pull down menu.



Unified creation tools

You can activate these tools also from the Tools/Unified pull down menu. The menu contains four basic creation tools: Curve, Sphere, Circle and Rectangle. With these four tools you can create all basic geometric shapes with a freeform, spherical, circular and rectangular shape.

Using the Rectangle tool

The unified Rectangle tool allows you to create various objects with a rectangular shape:

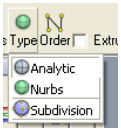
- Rectangular NURBS paths
- Analytic rectangular surfaces i.e. rectangles
- Extruded rectangles i.e. cubes
- Pyramids
- Pyramids with a cut top
- Rectangular freeform meshes
- Rectangular trim curves
- Paint rectangular shapes to bitmap textures
- Rectangular material mapping objects (parallel map, cube map, mesh map)

For example, to create an analytic rectangle:

1. Activate the unified Rectangle tool (Pull Down Menu/Tools/ Unified/Rectangle)



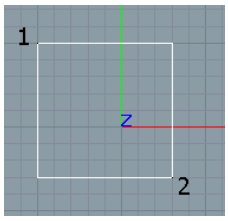
Unified Rectangle tool activated. The tool control bar shows you the unified Rectangle tool specific options



Select Analytic

2. Click the Type field and Select Analytic.

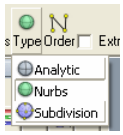
3. Now enter two points defining two opposite corners of the rectangle through the view window.



A rectangle defined by entering two opposite corner points

An analytic rectangle is very memory efficient and a fast object to render. It can be used for representing planar surfaces such as walls, floors, etc. However, it cannot be bent nor twisted. Analytic rectangles can only represent flat planar surfaces.

If you need to create a rectangle which can be deformed (for example, which can contain bumps in its geometry), you should create a rectangular NURBS mesh. This can be done as follows:



Set the Type field to Nurbs

1. Activate the unified Rectangle tool.

2. Set the Type field to Nurbs.

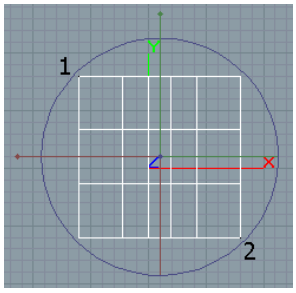
Now the tool control bar shows you options specific to NURBS meshes. For example, there are two resolution fields, which define the number of control points for the rectangular mesh.

3. Set U to 6 and V to 4 to create a Nurbs mesh with 6x4 control points.



Resolution set to 6x4

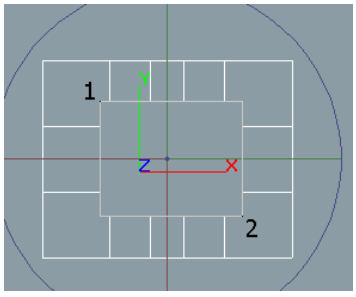
4. Enter two points through the view window to create a NURBS rectangle.



A Nurbs mesh consisting of 6x4 control points

The trim curve feature makes it possible to remove a part of a Nurbs surface. For example, to drill a rectangular hole to the mesh:

5. Make sure the Nurbs mesh is selected and activate the unified Rectangle tool. Set Class to Trim Curve and click two points over the Nurbs mesh to define a rectangular trim curve.



NURBS mesh with a rectangular hole

To create an analytic cube:

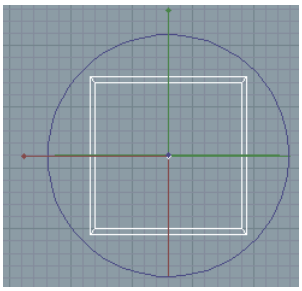
1. Activate the unified Rectangle tool, set the Type field to Analytic and set the Extrude option.



Options for creating an analytic cube

The Filled option is set by default. It means that the cube will be a solid one with 6 faces. If Filled is cleared, the cube will become hollow and without the front and back covers.

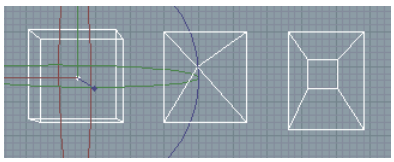
2. Define two opposite corner points through the view window to define the base rectangle. As soon as you enter the second point, the unified Rectangle tool starts to outline an apex point and the object looks like a pyramid. Don't define the third point. Just select Accept from the view's popup menu or from the tool control bar. A cube is created.



Analytic cube

If you want to create a pyramid instead of cube, enter three points and then press Accept. If you want to create a pyramid woth a flat top, enter four points.

In other words, the type of the object depends on the number of points entered. If only two points are entered, the result is a cube. Three points define a base rectangle and an apex (i.e. pyramid). Four points define a base rectangle and a top rectangle i.e. cut pyramid.



Three different objects achieved by using the Extrude option

Using the Circle tool

The unified Circle tool allows you to create:

- arcs
- disks
- cylinders
- cones
- cones with a cut peak
- hyperboloids
- disk and cylinder material mapping objects
- circular shapes to textures

To create an analytic cylinder:

1. Activate the unified Circle tool.



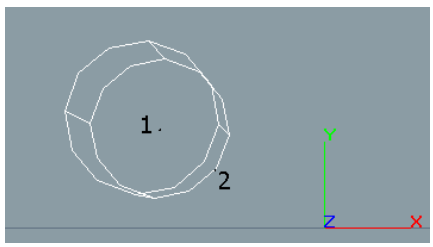
Options of analytic cylinder

2. Set Type to Analytic just like you did with the unified Rectangle tool and select Extruded from the Extrude gadget.

3. Click two points through the View window and select Accept. A cylinder is created.

If you accept the tool after three points, the result is a cone. Entering four points (center and radius for the bottom and top disks) creates a cut cone.

Note: the Define field allows you to control the creation method. For example, a circle can be defined by entering three points on it, by defining one center and one circle point or by defining a rectangle, which represents the bounding box of the circle.



Shaping a cylinder

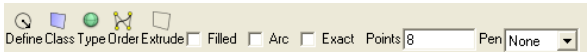
To create a circular Nurbs curve:

1. Activate the unified Circle tool.

2. Reset the Filled option (we don't want to create a surface).

3. Set the number of points defining the curve to the Points field, for example 8.

4. Enter two points through the View window to define the center and the radius for the curve.



Options for a circular Nurbs curve

Using the Curve tool

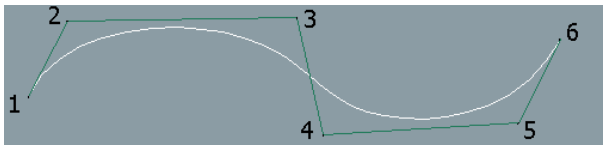
The unified Curve tool can create:

- freeform polygonal and smooth curves
- polygons, polyhedrons and polymids
- extruded curves
- meshes with a freeform boundary
- trim curves with an arbitrary shape
- freeform shapes to textures
- freeform material mapping objects

To create a smooth NURBS curve:

1. Activate the unified Curve tool

2. Enter a number of points through the view window. At your last point, select Accept from the toolbar or from the popup menu of the view window.



Drawing a smooth Nurbs curve

To create a polygonal curve:

1. Activate the unified Curve tool.



2. Set Order to Polygonal. Then define a desired number of points through the view window and click Accept.

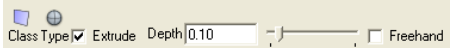
Polygonal curves have sharp edges - quadratic and cubic curves use control points which are interpolated through to create a smooth curve. You can control the sharpness of these curves by adjusting the weights of each control point.



Polygonal curve

To create an analytic polyhedron (extruded polygon):

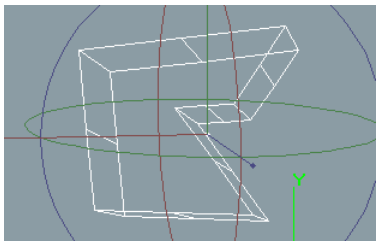
1. Activate the unified Curve tool



Options for a polyhedron

2. Set Type to Analytic and set Extrude. Define a desired number of points through the view window and click Accept.

3. The program starts visualizing the apex of a pyramid-like shape. A new mouse click would create such a pyramid, but choose Accept instead. In other words, Accept *twice* to create a polyhedron - one Accept + a new click creates a pyramid shaped polyhedron.



Polyhedron created

Using the Pen option to create surfaces

The unified curve tool supports a Pen option, which allows you to create Nurbs surfaces interactively. Use the pen option this way:

1. Draw a Nurbs curve as instructed earlier in this tutorial

2. Make sure the curve is selected and activate the unified Curve tool.

3. Select a suitable pen type from the pen gadget of the tool control bar.

4. Draw the new Nurbs curve. The program shows you the resulting surface as you enter the points of the new curve.

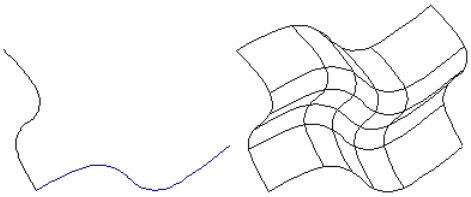
The following pen options are available:

None. If this is set, pen is though to be an infinitely small point and a curve is created.



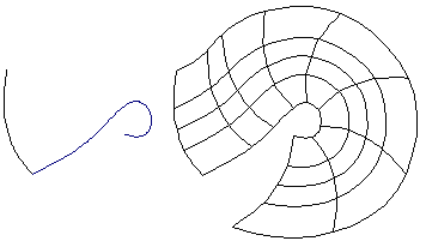
Plain curve

Rail. If this option is set, the selected pen curve will be swept along the curve drawn. The pen object is not rotated while sweeping.



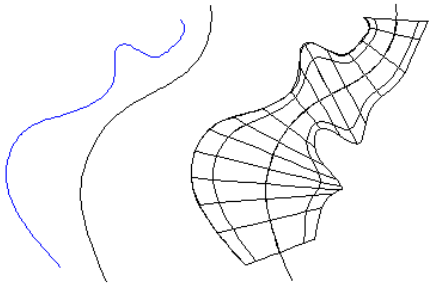
Rail pen

Sweep. This option is the same as the previous one except that the pen object is rotated according to the curvature of the curve drawn. For example, by using a circle as the pen object, one can create a circular tube.



Sweep pen

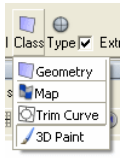
Rotate. This method uses the pen object as an axis and rotates the new curve around it.



Rotate pen

Using unified tools for mapping materials and for 3D painting

The unified tools can also create material mapping objects through the Class option. For example, we can use the unified Rectangle tool to map the wood material to a sphere using the Parallel mapping method.

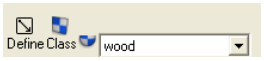


Class option of the unified Curve tool

To do this:

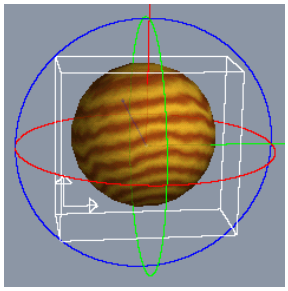
1. Create a test object, for example a sphere.

2. Activate the unified Rectangle tool and set Class to Parallel Map. Now the control bar contains a list of available materials. Select wood and define two opposite corner points through the view window.



The Parallel map option of the unified Rectangle tool

A parallel mapping object is created. You can render the view window to make sure the created material mapping object works.



Wood mapped sphere

Note that there are also many special purpose tools for creating mapping objects. You can find them from the Materials tab of the main toolbar, for example. If you need to map many materials, you might find the specialized tools more productive.

Similarly, you can find the 3D painting tools through the unified tool interface, although the Materials section of the toolbar contains more specialized painting tools.

Using Object Handles

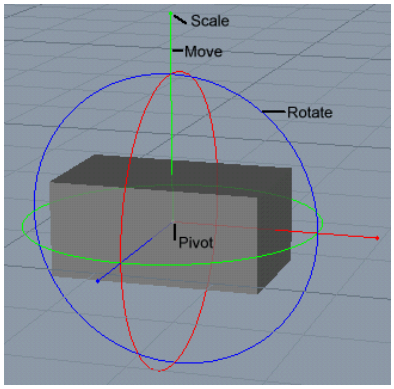
Object handles allow you to manipulate object properties directly - pretty much the same way people tend to manipulate objects in real life. For example, you do not have to push some 'move' button somewhere on your desk to enter 'move paper' mode or tool. You just grab the paper and move it.

Object handles in Realsoft 3D work the same way. They provide direct control over object properties like position, size and orientation.

Transformation handles

All geometric objects have nine transformation handles, which allow you to control position, size and orientation. When selecting multiple objects the handles of the first selected object are shown.

When you 'drag' (press the left mouse button while moving the mouse) one of the transformation handles, the corresponding transformation tool is activated. The transformation follows the mouse movement during the drag operation.



Transformation handles

Translation handles

To move a rectangle, follow these steps:

Create a small rectangle and select it. You can now see the three perpendicular lines, which are the handles to move the rectangle.

Move the mouse over the 'x' axis handle (the red axis), press the left mouse button and hold it. Note: when the mouse is positioned over the handle, the cursor changes to reflect the function of the handle. Also the status bar informs you about the purpose of the handle.

The rectangle is moved along its 'x' axis as you drag the mouse. When a desired new position is reached, release the mouse button to accept the tool. Hitting the 'Esc' key during the dragging cancels the operation.

Rotation handles

To rotate the rectangle just drag one of its circular rotation handles.

Scale handles

To scale the rectangle drag the end points of the move handles (the perpendicular lines). The scale handles are marked with small diamonds.

Note: the color of a handle corresponds to directions of the object space as follows:

- X - Red
- Y - Green
- Z - Blue

Activating handles by clicking them

You can also activate a transformation tool by clicking the corresponding object handle. In this case the clicked handle acts as a 'constraint' for the tool. All the tool options are shown and you will be able to use numeric interface, dragging and other snapping techniques.

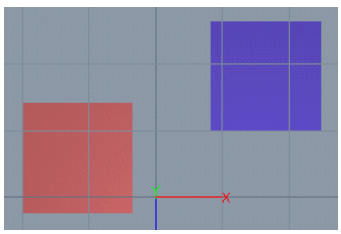
Let us see what is possible in this mode but first a quick excursion to a useful technique called 'point snapping':

To 'point snap' specific points of an object, move the mouse above and to the left of the point and press and hold the left mouse button. While holding the left button down move the mouse downwards and to the right (drag the mouse) until the point is inside the small rectangle that is drawn. Now release the mouse button. This handy operation is called 'point snapping' and can be used to snap the mouse position exactly to a desired point position.

Now we will perfectly align two rectangles using the "drag selection" technique:

Create two rectangles say 'red' and 'blue'.

Now we want to move the red rectangle so that the upper horizontal edge of the red rectangle is aligned to the lower horizontal edge of the blue rectangle.



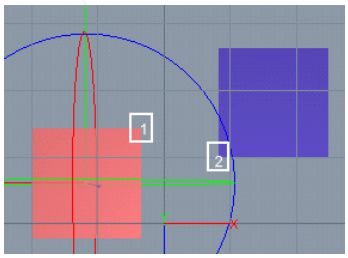
Rectangles to be aligned

Select the red rectangle and click the 'y' translation handle (the green axis). The 'y' handle becomes thicker to indicate that it is selected and the Move tool is activated. The rectangle now moves on its 'y' axis as you left click and move the mouse.

Drag select the top right corner of the red rectangle using point snapping [1]. Then drag one of the bottom corners of the blue rectangle to define the target position [2]. The two edges of the rectangles are now precisely aligned!

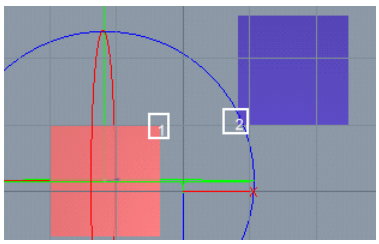
Now scale the red rectangle so that its top right corner matches the bottom left corner of the blue rectangle.

Click the end point of the 'x' translation handle (the red axis). Note: when you move the mouse over the end point handle, the cursor changes. This, in addition to the help text in the status bar, indicates that the scale operation will be activated if you click the mouse.



Snap the upper right corner of the red rectangle to the left bottom corner of the blue rectangle

Select the top right corner of the red rectangle using point snapping. Then drag the left bottom corner of the blue rectangle the same way. The right edge of the red rectangle is now perfectly aligned with the left edge of the blue rectangle!



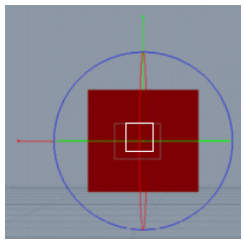
Use point snapping to define two points for exact result

The pivot point

When rotating and scaling objects, it is often necessary to define an appropriate center for the transformation.

When activating the Scale or Rotate tool from the Toolbar this center point is defined by the first mouse click. When using object handles the origin of the object space is used as the center of transformation. Many program call this 'center of transformation' point the *Pivot Point.*

To change the pivot point you can either drag the pivot point or click it. Dragging is perhaps faster to use but by clicking the pivot point you can use point snapping and other snapping methods for more precise results. Moving the pivot point means that you move the *object space origin*.

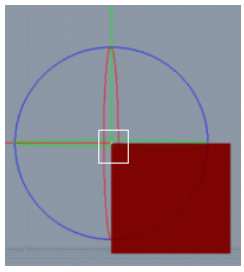

Drag over the pivot point

Now move the pivot point of the 'red' rectangle exactly to the left top corner:

Move the mouse over the pivot point handle and left click to activate the Move tool with 'Handles' option. Hitting the 'Insert' key also activates this "move pivot" mode.

Use point snapping to pick up the pivot point so that you can position it exactly at the desired point. Now the object handles move while you move the mouse.

Then drop the pivot by point snapping the top left corner of the rectangle. This accepts the Move tool and the pivot point is now located exactly at the top left corner of the rectangle.


Drag over the left top corner to drop the pivot point

Using modifiers with handles

The 'Shift' key is perhaps one of the most often needed modifiers. The purpose of this modifier depends on the tool.

Uniform scaling: If you scale an object, the Shift key acts as a '*uniform*' modifier. By dragging any of the three scale handles while holding down the Shift key, you can scale them uniformly in all the three dimensions. This is a common practice in many other applications as well, so it should be easy to remember.

For example, to make the red rectangle bigger:

Select the red rectangle and press down the 'Shift' key.

Drag a scale handle, while still holding down the Shift key.

The rectangle maintains its proportions in all three dimensions during the scale operation.

Rotating with angle grid When rotating an object, the Shift key activates the *angle grid.* The default angle grid is 45 degrees. This enables a rotation of, say, 90 degrees quickly and accurately.

For example, to rotate the red rectangle 90 degrees:

Hold down the Shift key

Move the mouse cursor over the blue circular rotation handle and drag (move the mouse while holding down the left mouse button) the mouse around the pivot point while still holding down the Shift key. You will only be able to rotate the rectangle in 45-degree steps and therefore 90-degree rotation can be easily achieved.

Duplicating and instancing: Many tools in the program use 'Ctrl' key as a 'copy' modifier. The 'Alt' key can be used for instancing objects. This is also true for object handles. Holding down the Ctrl key while dragging any of the handles first duplicates the selected objects and then applies the handle operation to the duplicated object. The original object is not affected.

Let's create four red rectangles.

Select the red rectangle (if not already selected) and press down the 'Ctrl' key.

Drag the 'x' move handle (the red axis handle) while pressing the 'Ctrl' key to duplicate the rectangle and move the new rectangle sideways. Repeat this operation three times to get three new rectangles.

There are many other useful modifiers which can be used for making object handles even more powerful. However, they are beyond the scope of this tutorial.

Using the numeric interface

It is often necessary to apply exact transformations to objects. Let us assume you want to rotate the red rectangle by exactly 23.4 degrees.

Make sure the numeric window is opened. If not, select the pull-down menu 'View/Numeric Window' to open it. You can now see how coordinates run on the numeric window as you move the mouse on the view window.

Select the red rectangle and double click one of its rotation handles. You can either double click the rotation handle or just click it two times. If you now move the mouse, you can see how the rotation angle field in the numeric interface shows you the current rotation angle.

Enter 'Numeric' mode by hitting the 'n' key. This activates the 'Angle' field in the numeric window. Alternatively, you can activate the 'Angle' gadget by clicking it with the mouse. Type the desired rotation angle of 23.4 degrees and hit the 'Enter' key to accept the rotation.

The selected red rectangle is rotated by exactly 23.4 degrees.

Also the Move and Scale tools support the numeric interface. However, when activated through handles, only one-dimensional scale and transformation factor needs to be defined. For example, to move the red rectangle 0.1 meters along its 'y' axis:

Click the green 'y' axis of the rectangle twice. The numeric window now shows you the displacement along the 'y' axis.

Hit the 'n' key and enter '0.1' in the Move1D field. Then press the 'Enter' key. The object is moved 0.1 meters along the axis.

Rotating and scaling about arbitrary points

Object handles allow you to scale and rotate objects easily provided that you want to apply the modification about the pivot point. As soon as you want to apply transformation about an arbitrary point, you have to move the pivot point. Moving pivot point frequently just to get an object rotated sounds like a lot of work and can soon become a 'major pain'. Fortunately, there is a faster way.

To rotate a rectangle about its left top corner:

Select the rectangle.

Click the desired rotation handle as usual. This activates the rotation tool about the pivot point. However, the tool activated by the rotation handle is the standard Rotate tool i.e. the very same tool you can activate through the toolbar or by hitting the 'r' shortcut key.

All transformation tools support 'Go Back' function, which allows you to re-enter any of the input points if needed. Therefore, it is possible to re-enter the rotation center. Select 'Go Back' from the view's pop-up menu or hit the 'Backspace' shortcut key.

You can now click or drag anywhere to define a new center of rotation. The transformation is still locked to the original, suitable rotation plane.

Transforming multiple objects

You can multi select any number of objects. In this case, only the first selected object provides the transformation handles so that the numerous handles don't clutter the entire view window. The handle of the first selected object transforms all the selected objects.

By controlling the selection order you can define a desired transformation space for modifying all the selected objects. For example, you can even add special 'reference' objects whose only purpose is to provide appropriate transformation spaces for the actual objects.

If you frequently have to transform objects in three special directions, just create three dummy objects whose object spaces match those directions. Then multi select the desired orientation object first and then the actual objects you wish to transform. The first selected object provides you conveniently oriented transformation handles.

Note: you can also change the object selection *after* you have activated a tool by clicking a handle. This is a very powerful feature, too. Let us demonstrate this with a simple example:

The goal is to rotate the red rectangle using the handles of the blue rectangle. You can, of course, move the pivot point of the red rectangle over the blue rectangle and then use the rotation handles to rotate the object. But this is not the same if the blue rectangle has a different object space orientation. However, there is an easy way to accomplish that: Just select the blue rectangle and click its rotation handle. This activates the Rotate tool and uses the pivot point of the blue rectangle. Then, using the select window, select the red rectangle. Then continue the rotation as usual. You have used a rotation handle of the 'blue' rectangle to transform the 'red' rectangle!

Accessing object specific handles

All geometric objects share a number of common properties, such as position, orientation and size. Correspondingly, they all support a set of common handles described above.

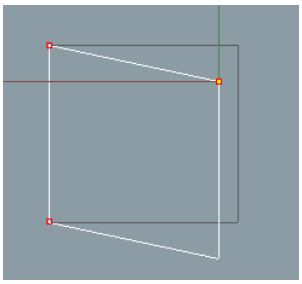
Additionally, geometric objects can define handles, which are specific to the object class in question. These object specific handles are shown when you enter the 'Edit' mode. Such special handles can activate arbitrary object specific tools.

For example:

Select the red rectangle.

Select 'Edit' from the view's popup menu or click the 'Edit' gadget in the tool control bar. The transformation handles disappear and three small point handles (little squares) appear.

Move the mouse over the handle in the top right corner. Press the left mouse button and drag the handle to a new position.



Editing a rectangle

Leave the edit mode by deactivating the 'edit' option from the popup menu or the tool control bar.

The object specific handles of a rectangle are the 3 points defining the geometry of the rectangle. You can use them to 'point edit' the rectangle. You can naturally also apply the usual tools like move, rotate, etc. and techniques like point snapping to these handles.

You can find more information about object specific handles from the tutorials describing geometric objects.

Working with Multiple Projects

Any number of projects can be created. Each project contains an independent set of materials, geometric primitives, post particle and post image effects. It should be noted that some data sections are shared between projects: render settings, grids, file render configurations are such shared objects.

You can cut or copy objects in one project and paste them into another project.

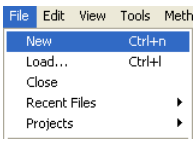
Animation time is also defined by a project: it is possible to continue modeling a project while playing the animation of another project.

There is one special purpose project called *material preview project*. By modifying this project, you can define how material preview renders preview icons for the select window's material tab.

Opening a new project

To create a new project, select the pull down menu: File/New.

This creates a new project and makes the new project the *current working project*. All newly created objects are inserted into the current project. Also, all windows (such as the Select window and the View window) reflect the contents of the current working project.



Create a new project

Switching between projects

For example, you can create objects in a temporary project and when ready, move the new object into the actual project by using the cut/paste tools.

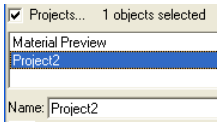
The current project

To make a project the current working project:

1. Click the Projects check box in the select window. The project list is shown.

2. Click the desired project to make it the current working project.

You can rename projects through the Name field.



Project list opened

Deleting projects

The current project can be deleted from the project list by selecting the pull down menu File/Close. All objects, such as materials, geometric primitives, etc. are deleted. Note: the files are not deleted from your harddrive. The previous project in the project list becomes the current working project.

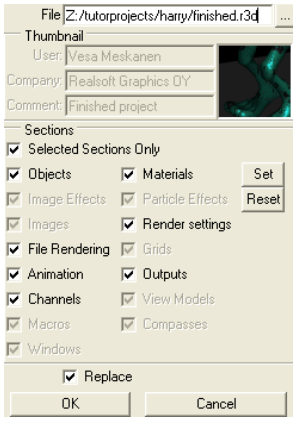
Loading projects from a file

The file load dialog, opened from File/Load pull down menu, is 'non modal' just like all the other dialogs in Realsoft 3D.

Load function automatically detects the format of the file to be loaded. If none of the installed file format classes recognize the given format 'Unknown file format' error message is shown.

The *Replace* check box defines whether the loaded objects should replace the existing objects or not. If not checked, new objects are inserted into the current project. If checked, existing objects are deleted from the current project before new objects are loaded in.

After a file has been selected, new format specific options appear to the window. The native Realsoft 3D file format provides options for loading individual data sections from a file. If the *Sections* box is checked, a suitable combination of data section selectors can be checked for loading only particular data from the file. For example, if you want to load only materials, check the Materials check box and reset all the other section check boxes.



The file load window in Sections mode, where you can select type of the data for loading

Saving projects to a file

The menu File/Save As opens a file dialog. This dialog, just like the File/Load dialog, is 'non modal'. It does not block use of other windows and hence it can be left open when necessary.

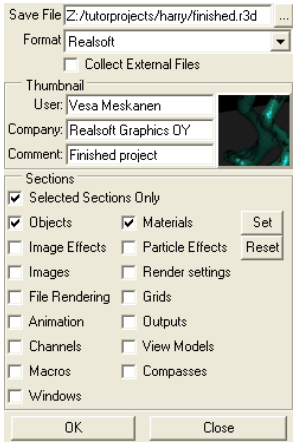
The *File Format* gadget defines the file format used for saving the project. The gadget shows the installed file formats, which support save operation.

The *Thumbnail* contains a small screenshot. To create a thumbnail picture, just click in the picture frame. Name of the *user* and *company* are filled automatically, but can be altered if needed. The *comment* field can be used to add some project specific information.

Sections defines, which sections in the current project should be saved. See the File/Load description above. If this field is set to Selected sections, the window is expanded and a number of check boxes are shown for section selection. Save all tries to save all data sections. Note that some formats may not support saving of all sections. You cannot save Realsoft 3D materials to a DXF file.

If *Collect External Files* is checked, all the bitmap texture files used by the scene are collected into one single textures folder, which is placed to the same folder where you save the project. This helps greatly when you need to move a project to another computer system. Finding all the textures (possibly scattered to many folders) would be laborious to do manually.

The pull down menu tool File/Save saves the current project using its name as a file name and the native file format. All data sections are saved.



The native Realsoft 3D file format selected. The lower part of the window allows you to specify file format specific options.

The pull down menu File/Save as Startup saves the entire current project to the file, which is automatically loaded in program startup. Use this function after you have customized the working environment to suit your personal needs. The backup copy of the original startup file can be found from the 'environments' folder.

Rendering projects

The current working project can be rendered to a file using the File/Render window.

This File Rendering window allows you to select file format, rendering settings, post effects, image resolution, frame command and file name for the image to be rendered. Both still images and animations can be rendered. You should insert a camera object to the project before rendering it to a file.

See the manual chapter Rendering for more information about this topic.

Snapping to Points Using Dragging

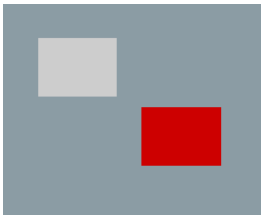
Many tools in the program require 3D geometry points as inputs. Typically, these geometry points are entered using the mouse, which is not a very accurate input device.

Snapping allows you to snap the entered mouse points to existing geometry points, achieving exact results during various modeling tasks.

In this tutorial we go through some basic snapping techniques, such as using the drag box and the lasso selector, as well as snapping to nearest point. You will find this functionality very useful in many modeling tasks.

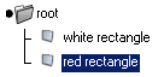
Dragging supported by the View window is a very powerful modeling aid. It allows you to snap to all existing points, pivot points, mid points and so on, all in one simple operation.

Consider the following situation: You have created two rectangles (red and white) and you have to align them so that the left top corner of the white rectangle matches exactly the right bottom corner of the red rectangle.



Initial position: two displaced rectangles

Do this:



1. Create two analytic rectangles as shown in the image.

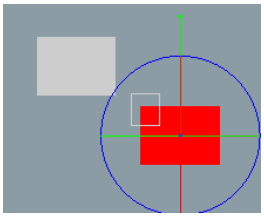
2. Select the red rectangle ...



... and activate the Move tool by hitting 'm' key or from the Transformation tab of the toolbar.

3. Move the mouse near the left top corner of the red rectangle. Press down the left mouse button and move the mouse. Drag box is shown. Move the mouse so that the left top corner is inside the drag box. Then release the mouse.

The corner of the rectangle was snapped into the mouse pointer. Now, move the mouse and the left top corner of the red rectangle follows the mouse precisely.

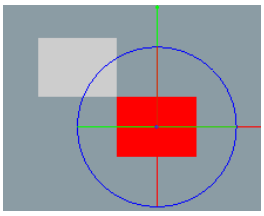


Drag over the left top corner of the red rectangle to snap to the corner

4. Now, drag over the right bottom corner of the white rectangle to snap the target point of the move tool there.

You can also drag over multiple points. In this case, all points inside the drag box are averaged. This allows you to snap to the middle point of the dragged points.

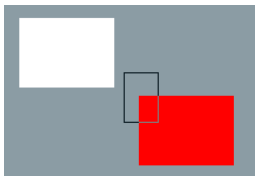
For example, let's move the left top edge of the red rectangle to the middle of the right vertical edge of the white rectangle.



Left top corner of the red rectangle snapped precisely to the right bottom corner of the white rectangle

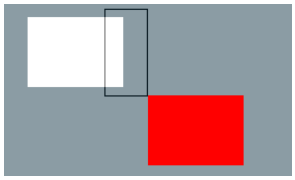
1. Select Edit/Undo from the pulldown menu to restore the original situation. Select the red rectangle and activate the Move tool.

2. Drag over the top left corner of the red rectangle. Move is activated and the dragged corner follows precisely the mouse.



Drag over the top left corner

3. Drag over the right vertical edge of the white rectangle so that its right edge is completely inside the drag box, and release the mouse.



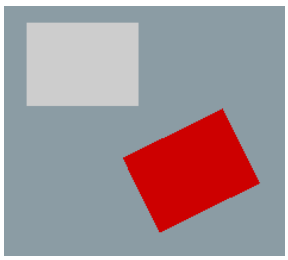
Drag over the vertical edge..



to snap to its middle point

Correspondingly, you can snap to the center of an object by dragging over the entire object. This averaging feature built in to the dragging makes the dragging very powerful feature because it allows you to achieve several important snapping methods through one easy to use interface. Dragging works with all transformation tools. Whenever you have to click the mouse in the view window, you can also snap to existing points using dragging.

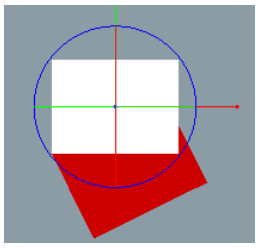
Let's see how you can take advantage of the dragging feature with the Rotate tool. The starting situation is like in the adjacent image. The task is to move and rotate the white rectangle so that its bottom edge matches exactly the top edge of the red rectangle. Thanks to dragging, it takes just a couple of mouse clicks to do this.



Initial positions

1. Select the white rectangle and activate the Move tool.

2. Now, using dragging, move the left bottom edge of the white rectangle to the left top edge of the red rectangle just like we did in the previous examples.



Corners snapped together

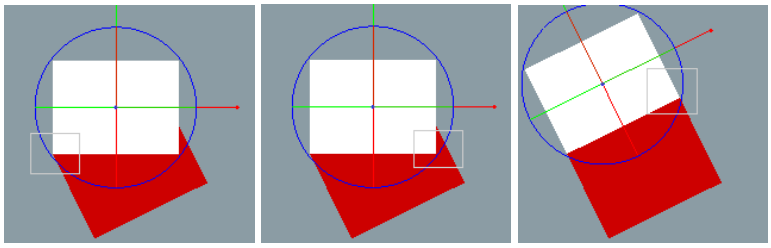
Next we should rotate the white rectangle about its left bottom corner to get the edges aligned.

1. Make sure the white rectangle is still selected and activate the Transformation/Rotate tool.

2. Drag over the left bottom edge of the white rectangle. This snaps the rotation center to the corner.

3. Enter the second point for the Rotate tool by dragging over the right bottom corner of the white rectangle. If you now move the mouse, the bottom edge points precisely towards the mouse.

4. Enter the last point for the Rotate tool by dragging over the right top corner of the red rectangle. Edges are now precisely aligned!



Drag three points for the Rotate tool: center, from point and to point



Note

Dragging averages *all* points inside the dragged area, including overlapping ones. For example, dragging over the white rectangle in the image above does not snap to the center of it, because the top edge of the red rectangle becomes included as well. The average of two points of the white top edge, two points of the white bottom edge and two points of the red top edge is not the middle point.

As you can see, dragging is an extremely simple but extremely powerful feature. With just a couple of mouse clicks you can move, rotate and scale about any point.

Lasso selector

In addition to the rectangular drag box, you can use a lasso selector to pick up geometry points. You can activate the lasso selector by holding down the lasso selector modifier. By default, this modifier is bound to the key '4'. Press down this key and drag on the view window to see the lasso selector.

3D snapping

Most people still use a two dimensional monitor and a mouse. This means that you can only see the position of a point using two coordinates x and y.

By default, dragging eliminates the third z (depth) coordinate so that when you, for example, move objects using dragging, their depth is not affected - the distance from view's camera to the two rectangles above did not change in the dragging operations.

By holding down a special *depth modifier* you can override the default behavior and dragged points are entered as true 3D points. The default key for depth modifier is ',' (comma).

Snapping to the nearest point

In addition to dragging, you can snap to single geometry points using snap to nearest modifier. Current mouse position is snapped to the nearest snap point when you hold down the key '5'. The nearest point snapping

works always in 3D (the depth changes as well). Note, that snapping starts as soon as you press down the left mouse button. This means that you can drag the mouse around and see which snap point becomes selected - release the mouse button when a suitable one is found.

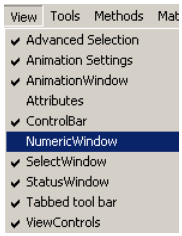
Snapping to the nearest curve

By holding down the key '6', you can project the current mouse position on the nearest curve point (any point, not only edge/control point).

Using the Numeric Interface for Accurate Modeling

In addition to the mouse, the input data for creating and transforming objects can also be defined using a numeric interface.

For example, to create a curve by entering the control points numerically:



Open the Numeric Window

1. Open the numeric window by selecting the pull-down menu View/NumericWindow.

2. Activate the Nurbs Curve tool from the 'NURBS' tab. The numeric window now shows you controls specific to the NURBS Curve tool. There are two fields: 'Position' and 'Relative'. The former allows you to define coordinates in 'world' space. The latter allows you to define points relative to the previously entered point (or to the 'hot point', which is the origin of the input plane, when no points has been entered yet). 'Relative' coordinates can also be accessed in polar (angle coordinate) form.

3. Hit the 'n' key. This activates the Position field. You can also click the field in the numeric window to activate it.



The numeric window with the 'Position' field activated

4. Enter three coordinates by separating them with a white space and hit Enter. This creates the first point for the curve.



Enter the first point and hit Enter

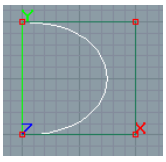
5. Enter three more points. After each entered point click the Enter key. Note: in some countries comma ',' is used as decimal point in which case you can replace the '.' in the following examples with comma.

```
0.1 0 0

0.1 0.1

0 0.1 0
```

Now four points has been added to the curve, and it has an arc-like shape.



Shape after 4 points

6. Let us try the relative measuring next: hit the Tab key. The Relative field becomes activated. Type the coordinates 0 -0.2 0 and hit Enter. A new point is added -0.2 meters below the first one. The relative measuring field remains active and you can continue entering coordinates this way, or hit Tab until the focus gets back to the Position field.

7. Now click the Accept button in the tool control bar. This terminates the curve tool and the curve is created.

Expression evaluation

All numeric fields support standard expression evaluation.

For example, to create a sphere with a radius sin(0.3)

1. Activate the sphere tool.

2. Click the left mouse button in the View window to define the center of the sphere.

3. Enter numeric mode by hitting the key 'n'. This automatically activates the Radius field in the numeric window. Type

```
sin(0.1)
```

and hit Enter.

This creates a sphere whose radius is 0.099833.

Note: Also vector fields support expression evaluation. For example, you can define the center of a sphere by entering three expressions separated by a space:

```
0.1+0.1 2/3 0.224*0.555
```

Measuring units and accuracy

The program uses internally the standard SI units. For example, positions and sizes of geometric objects are defined as meters. All numeric fields of the user interface convert internal SI units to desired current units.

You can select current units from the Preferences window.

If you want to work with millimeters:

1. Open the Preferences window from the pull down menu File/Preferences.

2. Go to the Metrics tab.

3. Set the Distance field to 'mm'.

4. Close the window.

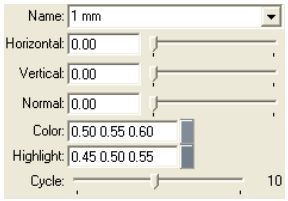
Now all numeric fields that represent distances (positions, lengths, etc.) are shown in millimeter units.

Note: you can change units any time. It has no effect to the scene you are modeling. The user interface only displays properties of objects using the new units. The Metrics tab of the options window contains also a slider for defining the number of decimals. If the measuring unit is meter and number of decimals is three, all dimensions are shown with millimeter accuracy (0.001 m = 1 mm).

Grids

In addition to dragging and the numeric interface, grids help you to enter accurate coordinates during the modeling process.

Any number of grids can be defined in the select window's grid tab.



Editing grid properties

To create a new grid object:

1. Go to the Select window's Grids tab.

2. Press the right mouse button and select New from the select window's pop-up menu.

3. Select Properties from the select windows pop-up menu to edit the grid properties. The following options are available:

- Name: The name of the grid object
- Horizontal, Vertical, Normal: sets the grid dimensions, which define the distance between snap positions. Usually, all three fields are set to the same value. If you wish to enter small values like 0,001, make sure the value of the options/window/metrics tab is set to match.
- Color: The base color used for grid drawing.
- Highlight: A second color used for drawing the grid lines, whose frequency is determined using the Cycle slider.
- Cycle: Defines the density of grid lines that are drawn with the highlight color.
- Dynamic: Automatic adjustement of snapping interval.
- Dynamic Color: Automatic adjustement of grid color for dynamic grids. The shorter the snap interval, the lighter grid is drawn.

You can activate the desired grid by dragging and dropping it into the view window. This sets the current grid for the view window and activates both Snap to grid and Show grid options. You can also use view property window's Input tab for controlling view's grid snap and draw options.

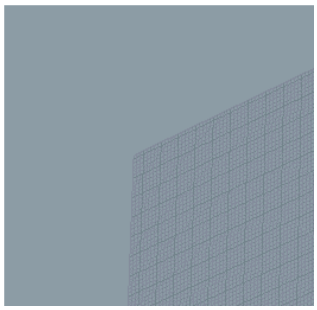
Note: do not mix the grid with the ground plane visualization in perspective views. The ground plane has nothing to do with snapping and is provided as an orientation guide only. It is automatically turned off when grid drawing is enabled.

The view control bar allows you to switch Snap to grid and Show grid options on/off quickly any time.

Show Grid and Snap to Grid icons

The grid drawing fills the entire view window. If you zoom out the view window so that the grid appears denser and denser, the grid system stops drawing the base colored lines at some point, because the too densely drawn lines would not serve any useful purpose. Drawing only the highlighted lines keeps grid visualization useful. If you continue zooming out, also highlighted lines get too dense and their drawing will be stopped.

The same density control is used when perspective projection is enabled and view camera is not locked to the grid plane. Only the part of the infinite grid plane, which is not too dense, is visualized.


Too dense part of the grid is not drawn in perspective mode

The automatic density control does not affect grid snapping, only grid drawing.

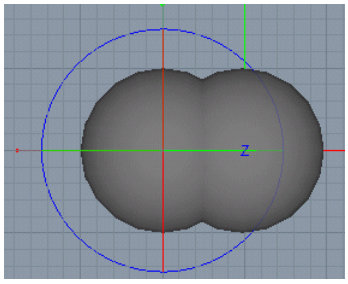
Boolean Operations

Boolean operations offer a very powerful way of creating and modifying objects.

Consider a situation where an engineer cuts an object made of pine with a tool that has just been painted. Since the paint on the tool is still wet, it sticks on the object and the cut surface is painted with the wet paint.

The engineer notices this and replaces the tool with another clean tool. The new cut surface is clean showing the beautiful texture of the pine. The Boolean operations of Realsoft 3D work in a similar way.

A simple example of a object created with a Boolean operation is a lens. A lens can be thought of as the intersection of two partially intersecting spheres.



Two intersecting spheres

Tutorial level: Beginner

Example project: 'tutorprojects/modeling/boolean/splitlens'

1. Create an analytic sphere using the Analytic Sphere tool.

2. Duplicate the sphere by holding down the Ctrl key while moving it so that the original and duplicated spheres are partially intersecting (you can also use Edit/Duplicate pull-down menu to duplicate an object).

3. Multi select both spheres. As usual, this can be done by clicking the both spheres in the select window while holding down the Shift key.

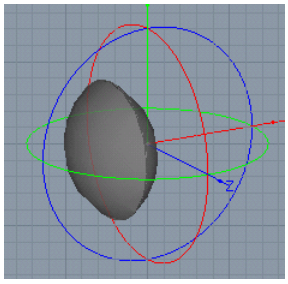


Intersect tool activated

4. Switch to the Boolean tab of the toolbar.

5. Activate the second tool, Intersect.

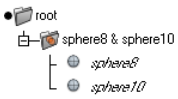
6. Then click Accept.



Intersection of two spheres

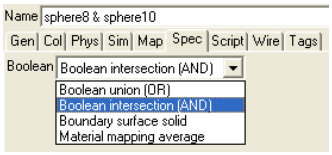
The resulting object will be the part of the work piece that is inside the tool. It is a lens. If you change the material of the lens to glass, it can be used as a magnifying glass!

In the select window, you can see that the two objects have been replaced with a new level named 'sphereX & sphereY'.



Structure of a Boolean operation between two spheres

This reveals the nature of Boolean operations in Realsoft 3D: a Boolean operation is just a *level* object consisting of objects to be operated. The objects used in these operations can be complex ones with many levels of hierarchy, and they can contain objects that result from previous Boolean operations.



Activating intersection for a level object

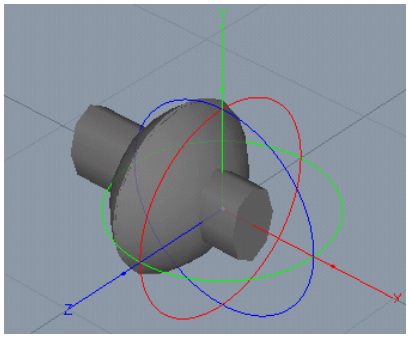
This means that instead of using the procedure described above, you can also make Boolean operations manually. When you create a new level, its type is set to union (OR). If you want to do intersection instead, open the property window and change the Type field of the Spec tab to Boolean Intersection.

Note that the original Boolean operated objects (sphere1 and sphere2) can be modified after the operation just like any regular objects. You can even activate the Simulation for the Boolean level and let gravity and other simulation effects to animate the sub objects.

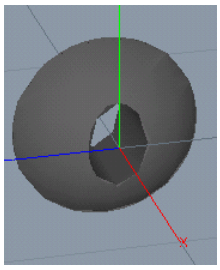
Let's continue experimenting with the lens object by cutting an oval hole through the lens.

7. Rename the created intersection level as Lens.

8. Create an oval analytic cylinder passing through the lens object.



A cylinder passing through the lens object



A lens with an oval hole

9. Select the lens and the cylinder object (in this order) and activate the Boolean operation tool again. Select the third Boolean operation option Remove and click Accept.

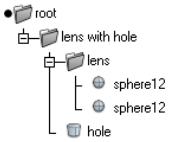
A new object 'Lens & !cylinder' is created. Rename it as 'lens with a hole'.

In Realsoft 3D it is possible to execute the Boolean operations so that the tool affects not only the volume but also the surface of the object that it is operating on. This is a very useful feature. For example, if you cut a notch in a pine board using a shiny reflecting (material attributes) tool and use Remove operation with Paint option, you get a shiny reflecting notch on the board.

Likewise, if you use a red tool you get a red notch. And if you use a chimney to cut a notch, you get a piece of wood with a notch of brick coating. On the other hand, if you use any of the non-painting operations, you get a clean notch of pine on your pine board (the wood texture of the pine board 'x-rays' right through the object so that you get a continuity of the texture). In this case the surface of the tool does not affect the object that is operated on.

To test this feature:

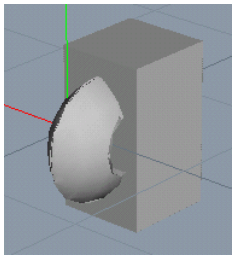
10. Open the 'Lens with a hole' object by clicking its '+' handle in the select window. You can see that the level consists of two sub objects: the original complete lens and the cylinder representing the oval hole.



A hierarchical Boolean operation

11. Select the oval hole object and open the property window. Go to the Col tab and change the color of the cylinder to red (1 0 0).

Click Render and you will see a white lens with a red hole.



A cube enclosing half of the lens

You can disable this Paint effect by clearing the Paint in Boolean Operations option in the cylinders property window's 'Gen' tab. You can obtain the same results by setting the Ignore materials option of the Boolean tool, instead of changing the attributes afterwards.

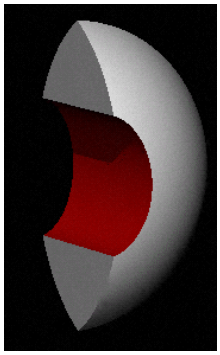
Finally, we can make a section diagram of the lens by removing half of the lens using a cube object.

12. Create an analytic cube so that half of the lens is inside the cube.

13. Select the 'lens with a hole' and the cube. Then click the Boolean tool in the toolbar, set Intersection option and click Accept.

If you have OpenGL switched on, the inaccurate polygonal representation might not look that beautiful. However, if you ray trace the view, you can see a perfect half of a lens with an oval red hole through it.

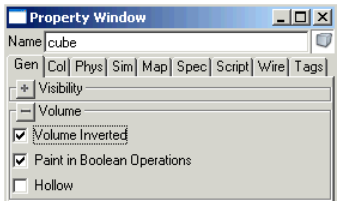
This is still a very simple object for Realsoft 3D. You can continue Boolean operations resulting in more complex shapes. They can be saved to disk and used in constructing other objects. As you complete useful new objects and tools, using the program becomes easier and faster.



A split lens with an oval hole through it

Inverting volumes

We created half of lens as an *Intersection* of the lens object and a cube. All parts of the lens outside the cube were removed.



Inverting volume

If we invert the *Volume* property of the cube, the operation will cut the part of the lens inside the cube. As a matter of fact, this is the way the Remove Boolean operation works. *Remove a cylinder from a lens* is the same operation as *Intersect a Lens with an inverted cylinder*.

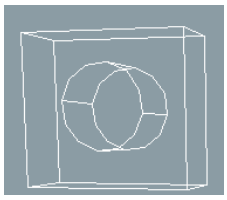
To invert the volume of the cube, which cuts half of the lens:

14. Select the Cube and open the property window. Go to the Gen tab and set the Volume Inverted checkbox.

The previously removed half of the lens becomes now the result of the operation.

Wireframes of Booleans

Real-time visualization of a Boolean operation does not always show properly the true shape of the object. The reason for this is that the wireframe for a Boolean operation level is computed from wire frames of sub objects, and performing the operation often increases the quality requirements. Wire frame and shaded quality can be adjusted using the quality settings of the property window's Wire tab.

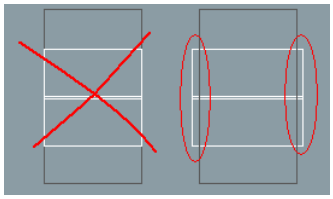


Parts for a Boolean operation

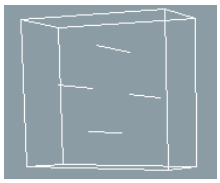
The automatic computation of Boolean wireframe is not always desirable. Turning Boolean wireframe computation off can increase the screen refresh rate and sometimes even give a better visual look. The next example demonstrates this.

1. Create an analytic cube and an analytic cylinder through it.

2. Now check an important thing: take a side view and make sure that the cylinder is long enough. The ends of the cylinder must stick out from the cube. If the objects are equally long, drilling with the cylinder will not succeed. Stretch the cylinder, using the blue rotation handle, if necessary to achieve this.



Verify that the drill is long enough

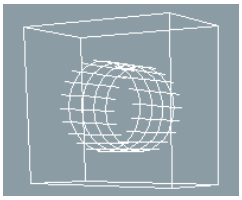


Insufficient Boolean wireframe

3. Select the cube and then (with the Shift key held down) the cylinder. Activate the Boolean tool and select the third operation type Remove. Click Accept.

All the parts of the cylinder wireframe outside the cube are cut away (because they do not belong to the resulting volume). This leaves very little wireframe left. Let's first adjust it by using the wireframe quality controls.

4. Open the hierarchy of the level 'cube & !cylinder' created by the Boolean operation from the select window. Select the cylinder. Open the property window and go to the Wire tab. Change the Wire density slider from the default value 1 to the value 5. Then select the parent level. Click AND wireframe off button first on and then off to cause a refresh of the Boolean wireframe.



Cylinder wireframe density increased

Now the denser wireframe clearly shows that a hole has been drilled through the cube. However, the original wireframes of the cube and the cylinder would give an elegant and beautiful representation of the operation (see the illustration of step1 above). To achieve this:

5. Select the cylinder again and drop the Wire density value back to one. Then select its parent level and check the AND wireframe off box. Now the object disappears from the view window. This happens, because the Boolean tool made sub objects invisible, so that they would not obscure the computed Boolean wireframe. Switch to the Gen tab of the property window. Select both the cube and the cylinder, and clear the Invisible in real time rendering box. The original wireframes appear again, which is what we wanted.

Summary:

- A Boolean operation level automatically computes a new wireframe/shaded representation from its child objects
- The quality of the Boolean visualization can be controlled using the Wire attributes
- Boolean wireframes can be switched off if necessary

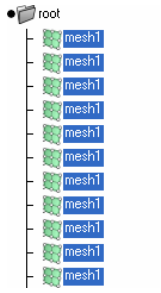
Boundary representation solids

In the previous examples we used analytic sphere, cylinder and cube objects in Boolean operations. All analytic objects are very fast in Boolean operations because their interior volume is defined efficiently by a mathematical formula.

However, it is also possible to use one or more surfaces, such as NURBS or subdivision meshes, to construct new solid objects. This kind of freeform solids are defined using so called Boundary Representation Theory. Solids defined this way can be used in Boolean operations the same way as analytic objects.

To create a new solid object from a set of NURBS meshes:

1. Create and select any number of surfaces defining a closed volume.



Two nurbs meshes and a set of weld objects defining a closed volume selected



Note

Closed volume means that it should not be possible to get inside the object without passing through a surface.

2. Activate the Boolean tool. Select the last operation Solid and click Accept.



Operation set to Solid

This creates a new solid primitive from the selected surfaces. Now you can use this new level object similar to other solid objects.

For example, to drill a hole through the new boundary solid object:

3. Create an analytic cylinder through the object.

4. Select the level object and the cylinder (in this order) and activate the Boolean tool with Remove option. Then click Accept and render the view to see the accurate result.

Editing Boolean operations

When you execute a Boolean operation, the hierarchical structure of the object is modified to reflect the resulting object. If you move the lens object of the previous example, you notice that the hole faithfully follows the lens.

On the other hand, you can select the hole and make it bigger. The result of the Boolean operation changes immediately to reflect this change. You can also create animations where the hole moves, or where the progression of the hole being drilled is animated. The hierarchical implementation of Boolean operations makes this very easy in Realsoft 3D.

Advanced Boolean Operations

Realsoft 3D has a very powerful Boolean feature set that allows the user to create subtractions and unions of objects in many ways. Not limited to simply cutting a hole through an object, Realsoft 3D can cut entire scenes in half and have complex level structures join together to create very detailed objects.

Most commonly, Boolean operations are used to subtract one object from another. For example, a drill cutting a hole into a desk. But the power of Booleans allows us to use more than just one object. By dropping a whole series of objects to a level, we can make multiple cuts into one object:

1. Create an analytical cube.

2. Duplicate the cube (Ctrl+d) and size it so it is a very thin slit, smaller than the original using the handles. Position it so that it is slightly inside the original cube.

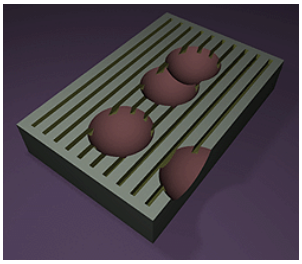
3. Drop this thin cube into its own level (Right click "drop to a Level") and rename the level: "Cuts"

4. Select the thin cube in the level. Make the "cuts" level the current level (Right click "Make Current")

5. Turn on Macros/Record (Macro pull down menu).

6. Duplicate the thin cube and move it up slightly. (Do not use the handles to duplicate - use either the pull down menu, right click, or the keyboard shortcut Ctrl+d to duplicate)

7. Turn off Macros/Record. Press the Ctrl+g key to execute the macro once. Repeat until you have the desired number of thin cuts (say 10).



8. Make the Root level the current level (Right click "Make Current"). Select the first cube, and the level "Cuts".

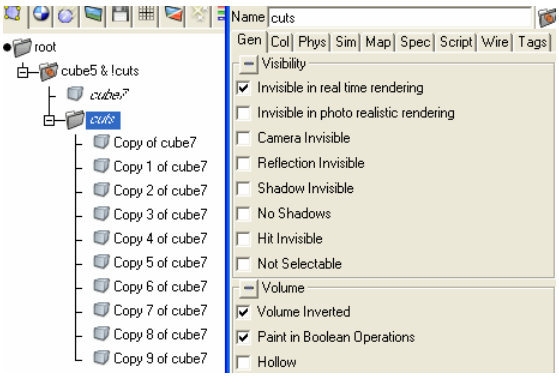
9. Select the Boolean tab of the toolbar.

10. Select the 'Remove' icon. Leave all other values to default and accept.

The Boolean operation now cuts the thin cubes from the original cube. This is dependent on the selection order - always select the object you want to cut FROM, then the object to be cut OUT.

Note that we can now add object freely to this "Cut" level and they all cut out from the original cube. Typically, if only 2 objects are used in a Boolean operation, the Cut object is set the attributes: invisible in wire, and volume inverted.

By dropping these objects to a level before applying the Boolean operation, the level inherits these attributes and all objects underneath inherit the level's properties. Hence we can also change the level's color to change all the cut colors. This ability to add objects to the level without having to set the Inverted volume attribute allows us to take this example further:



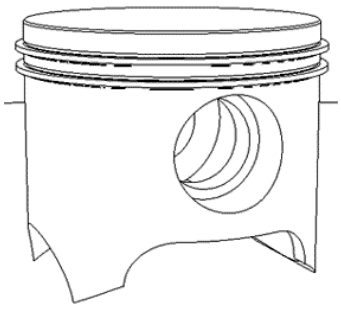
Adding materials to a cut object level allows us to create all sorts of interesting effects - For example, having a brick outer wall, then cutting holes in the wall with wood mapped cubes would yield a wooden window sill in the walls.

It is important to note that Booleans can include other Boolean objects. To create a piston ring, one creates a cylinder, cuts a hole in the middle, then uses that object to cut out from another cylinder, then duplicating the first cut ring to create a piston with a ring cut and piston ring inside. Such nesting of Booleans is limitless - cutting and shaping an object until the final form is realized.

Primarily analytical objects are used as they are closed shapes and the volume is easily inverted. The polyhedron shape is of particular use - it is an 'analytical freeform' shape. By using polyhedrons creatively, one can create almost any shape.

Attached is a relatively simple scene making good use of Boolean operators. Starting from a simple cylinder we "chiseled" out the form of a piston. Thinking in terms of subtraction is the key - How you want to "Cut" the object to obtain the shape you desire. Often this means cutting the cut tool - as with this scene. Examine how chamfers were cut from the shape with cones, then cut again from the piston cylinder. Note that this object was created solely from analytical objects - predominantly cylinders.





A useful technique for modeling complex forms such as this is to duplicate many of the objects from the original cylinder. This creates objects in the correct transform space and often only requires simple scaling or rotating.

When attempting to visualize the shapes inside a Boolean - creating the cut shapes OUT of the Boolean level is useful, placing them in the correct location, then dragging the icon into the "Cut" level.

Boolean representations can often be quite confusing and the Wire and Open GL representations quite inaccurate. Remember that these are merely previews of the object - renderings show the final form. Working in wireframe is by far the fastest method with Booleans - as the Open GL view is constantly being updated, computation times can sometimes be too slow for complex Booleans. Working in wireframe is best with complex scenes.

Under the object properties/wire tab of a Boolean level is a tag "AND wireframe off". This can be set to turn off the automatic Boolean wireframe representation if needed.

Problem solving Booleans

Occasionally, after much cutting and pasting between levels - errors can appear. These errors can either be strange artifacting - which is quite possibly the result of two shapes exactly on top of each other. Or Booleans looking completely wrong. If this is the case, the easiest method to check your objects is to drag half of the objects out of the Boolean level and render (Often the wireframe representation is enough). If the Boolean looks OK then slowly add various shapes back under the Boolean level until you can pinpoint which shape is causing the problems (drag the icons out of the Boolean level and turn off render.) The problems are most likely due to having 'Invert Volume' on. Check the problem objects for unusual attributes under the 'Object properties/Gen' tab.

One must be very careful with the placement of cutting objects. Sometimes duplicates may appear unnoticed by the user and strange artifacting may result as two identical objects may be cutting another. The finite accuracy of computers allows us to place objects in exactly the same positions, but when cutting two objects, it is imperative that the cut object is slightly away from the edge or corner to allow at least some cut or no cut at all. Otherwise strange results might appear. This is also applicable when mapping scope or finite materials.

When mapping Boolean objects, it is often a good idea to drop the Boolean level to another level, then apply the material.

Lattice Mapping

The lattice mapping system allows you to attach an object to another object called *lattice*. When the lattice object is modified, also the attached object becomes modified accordingly.

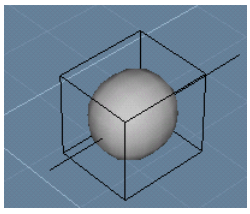
Any object can act as a lattice object. For example, you can attach a number of logos to a NURBS curve and deform the curve to control the positions of the logo objects.

The idea is that a complex geometric object, such as body of a character, can be controlled easily using a simpler object, such as a skeleton.

Tutorial level: Medium

1. Create a SDS sphere.

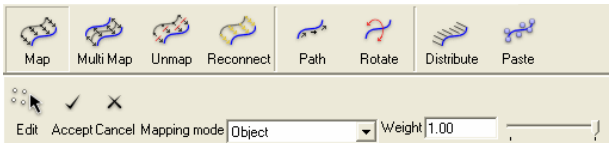
2. Create a horizontal straight NURBS curve consisting of four control points and passing through the sphere.



A NURBS curve passing through a SDS sphere

3. Multi select both objects, the SDS sphere first, the curve second. Make sure that you select the objects in this order because the Mapping tool maps the first selected object to the last selected object.

4. Click the Map tool in the Lattice Mapping tab of the toolbar. You can now see a couple of mapping specific options in the control bar. These default options are just fine, so Accept the tool.



Apply the Map tool to the selected sphere and curve objects

We have now mapped the sphere to the curve. Select the curve and move it. The sphere is moved with the curve. If you rotate the curve, also the sphere gets rotated. How about if you want the curve to affect only the position of the sphere, so that the curve does not affect the direction of the sphere in any way? To do this:

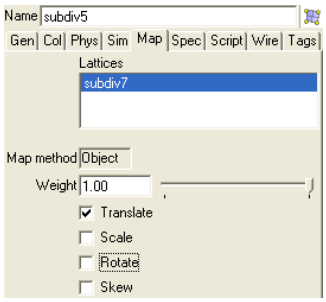
1. Select the SDS sphere.

2. Open the property window and go to the Map tab.

3. Reset the Rotate check box.

If you now rotate the NURBS curve, the sphere is only moved. In other words, the curve controls only the position of the sphere.

If you want to use the curve to control the orientation of the sphere, not the position, just reset the Translate check box and set the Rotate check box.

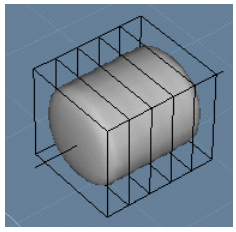


Rotate check box reset. The curve affects only the position of the sphere.

Pointwise mapping

In the previous example, we mapped a sphere to a curve using object mapping option. Therefore, the lattice curve controlled the four properties of the object space of the sphere: position, size, orientation and skew.

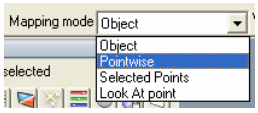
To allow lattices to fully deform objects, point mapping should be used instead of object mapping.



A NURBS curve passing through a SDS cylinder

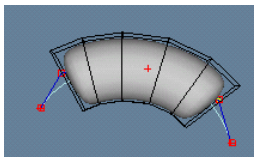
To map an object to a curve using pointwise mapping:

1. Create a freeform object (such as a SDS cylinder, which containse enough points to allow interesting deformations). Create also a NURBS curve passing through the mesh just like we did in the previous example.



Use Pointwise mode

2. Select both the cylinder and the curve in that order. Click the Map tool. Now you can again see the mapping options in the Control Bar. Set the Mapping mode field to Pointwise and accept the tool.


NURBS curve bent. The mapped SDS cylinder is bent accordingly

3. Now select the curve and enter the Edit mode. Point edit the NURBS curve. The mapped mesh is deformed with the curve.

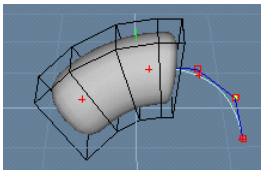
Modifying mapped objects

If the position of the mapped object is not correct, you can simply move it into the new position. For example:

1. Select the SDS cylinder we created in the previous example and turn off the Edit mode if it is still on.

2. Activate the Move tool.

3. Set Target=LatticeCoords from the tool control bar.


SDS object moved along the curve

4. Click a point on the lattice curve. When you move the mouse, you can see how the SDS cylinder moves along the curve. When a desired position is found, click the mouse again.



Note

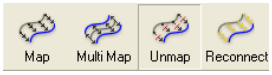
When moving mapped objects, the first point entered for the Move tool is very critical. Make sure you click over the curve so that the move tool has a good reference point to start moving the target along the curve. If the first point for the move tool is heavily displaced from the curve, the moving route in the curved lattice space is not intuitive, making the operation hard to control.

Detaching mapped objects from lattices

You can unmap an object from a lattice object by using the Unmap tool. For example, to detach the SDS cylinder from the curve.

1. Select the SDS cylinder and the curve objects (in this order).

2. Activate the Unmap tool and click the Accept button in the tool control bar.



Unmap tool activated

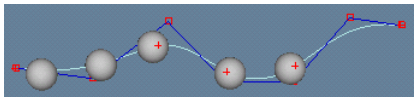
If you now modify the curve, the SDS cylinder is not affected in any way.

Mapping multiple objects to a lattice

You can map any number of objects to a single lattice object.

There are two ways to accomplish this. First, you can multi select all the objects and map them to the desired lattice object at once. Just make sure the lattice object is the last selected object when accepting the Map tool.

Or, you can select and map objects to a lattice object one by one. You can repeat the Map tool any number of times for the same lattice object.


A NURBS curve passing through five spheres

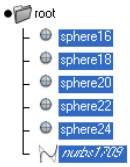
Let's map a number of spheres to a curve.

1. Create five small analytic spheres and a NURBS curve passing through the spheres.

2. Hold down the Shift (multi selection) key and multi select all objects by clicking their names on the select window one by one: spheres first and the curve last.

3. Activate and accept the Map tool with the default options.

Modify the NURBS curve. The spheres are modified with it.



Select the NURBS curve last by Shift-clicking the objects from top to bottom

Swapping lattice objects

The Reconnect tool detaches objects from their current lattices and attaches them to a new lattice object.

For example, a sphere moving along a NURBS curve can be reconnected to a skeleton to get the sphere to move along the skeleton.

Note that only the lattice object is switched. All other mapping information remains untouched.

This tool can be very useful when you need to map complex objects to curved lattices. For example, you might want to map a NURBS mesh to move along a cylinder surface. Instead of trying to bend the mesh to match the shape of the cylinder, simply map the mesh to a planar surface and animate it there. When the animation is ready, reconnect the animated mesh to its actual cylinder lattice.

Let's see how this works.

1. Create a NURBS mesh (rectangle) consisting of 5 x 5 points and map it to a bigger analytic rectangle using pointwise mapping.

2. Activate animation recording and move the mesh in the initial position in the beginning of the animation.

3. Move the time slider to the last frame and move the mesh to another position. You can also single point edit the mesh for animated deformations.

4. Reset animation recording and play the animation to see how the NURBS mesh moves in the space defined by the analytic rectangle.

5. Now, create an analytic cylinder (actually a NURBS cylinder or sphere or any other geometric object with a curved surface will do).

6. Select the animated NURBS mesh and the new cylinder, activate the Lattice Mapping/Reconnect tool and click Accept.

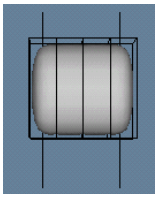
Play the animation to see how the mesh moves along the surface of the cylinder.

Multi mapping

The previous examples used only one lattice object. However, a single object can be mapped to any number of lattice objects.

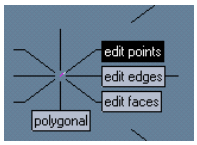
For example, you can use multiple NURBS curves to control different parts of a single SDS mesh.

Let's see how this work. Let's create a SDS cylinder whose left side is mapped to one curve and the right side is mapped to another curve.



A SDS cylinder and two vertical NURBS curves

1. Create a horizontal SDS cylinder (use for example 4*8 points) and two vertical NURBS curves at both ends of the SDS cylinder.

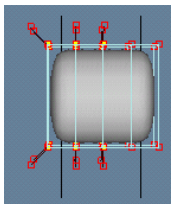


Enter point editing mode

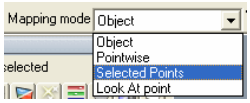
2. Select the SDS cylinder and enter point editing mode (select Edit Points from the compass).

3. Select all the left side points of the SDS cylinder using dragging.

4. Multi select the SDS cylinder (with its left end points selected) and the left side NURBS curve in this order.



Select points in the left half



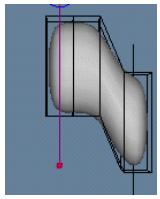
Apply the Map tool with the Selected points mode

If you look at the View window now, you can't see any points selected in the SDS cylinder. This is normal. The points are still selected, although not highlighted because of multi-selection situation.

5. Activate the Map tool and set the Mapping mode option to Selected points. Then Accept the tool.

Try moving the left side NURBS curve now. The left end of the SDS cylinder is moved with the curve.

6. Now repeat the steps 3...5 for the right side points of the SDS cylinder. In other words, select the right side points of the SDS cylinder, then multi select the right side NURBS curve and apply the Map tool again with the Selected points option.



The leftmost curve moved upwards

Weighted mapping

The Weight parameter controls how strongly lattice objects affect the mapped points or objects. This is often needed in multi mapping.

For example, you can map a point on two lattices so that the first lattice affects the point with 30% strength and the other one affects the point with 70% strength. Or you can define weights so that the shorter the distance to a skeleton, the stronger the skeleton affects the points/objects.



Mapping with 50% strength

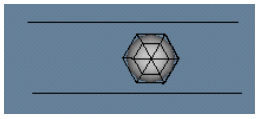
Let's multi map a SDS sphere to two NURBS curves and experiment with weights:

1. Create a SDS sphere and two identical NURBS curves passing through the sphere (duplicate the curve to get two identical curves).

2. Select the sphere and the first curve just like we did in the previous examples and apply the Map tool using the Object mapping mode.

3. Map the sphere on the second curve.

You have now mapped the sphere on two curves using equal full weights.



A sphere multi mapped on two NURBS curves with 1.0 weights

4. Now move the curves. The sphere is always located exactly between the NURBS curves, because both curves affect the sphere with an equal weight.

To make the first curve affect the sphere only 10%:

5. Select the sphere, open the property window and go to the Map tab.

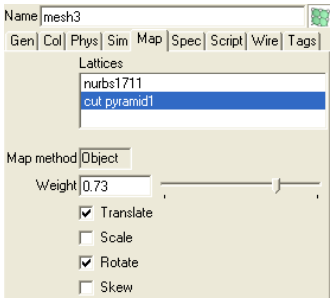
6. The Lattice field shows you the name of the first curve. Set Weight to 0.1. The sphere is moved towards the second curve.

7. Now, pick the name of the second curve from the property window's lattices list. Then set also the weight of the second lattice curve to 0.1. The sphere is moved back to the center of the curves because both curves affect again the mapped object with equal (0.1) weights!

Weights are like strings, which attempt to pull the mapped points to the position defined by the lattice objects. The higher the weight, the stronger the string. Therefore, if you have only one lattice object, weights are irrelevant, because even the weakest string will pull the mapped object or point. There is no other string resisting it.

Using the property window to edit mapping parameters

The Map tab in the Property window controls mapping parameters.



Lattice mapping properties

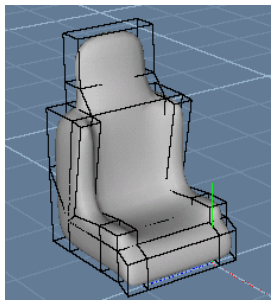
- *Lattices:* The list of lattice objects controlling the selected object. Other gadgets of the tab display to properties related to the lattice defined by this gadget.
- *Map method:* Displays the type of the lattice mapping (object or pointwise)
- *Weight:* The control weight of the lattice. In pointwise mapping, each point can have an individual weight, and the gadget shows and changes the weight of the selected points (set the mapped object to edit mode if you want to change point weights).
- *Translate, Scale, Rotate, Skew:* These gadgets define which components of the target object's space are affected by Object type lattice mapping.

Using the vertex painting tool to control weights

You can use the Vertex Painting tool (this multi-purpose tool is located in the Materials tab) to control weights of pointwise lattice mapping. Select the mapped object and then activate the tool. The tool control bar shows a selector Property to be painted. Pick Weight from it, select a suitable brush size, define the weight and paint over the desired points.

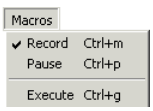
Recording and Executing Macros

When you use the program in the macro recording mode, all actions are recorded into the *current macro*. Current macro can then be executed, for example, to speed up a certain modeling task. You can also rename the current macro to create *a named macro* and bind the named macro to keys, buttons or menus.


A chair for a train

This tutorial shows how macros can be recorded and executed. The GUI chapter of the manual explains how to bind macros to the components of the user interface.

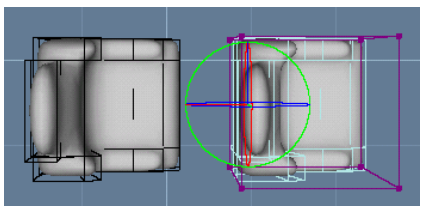
In this tutorial, we create one chair and record a macro, which duplicates and moves the chair. Then we execute the macro a couple of times to get a row of chairs.



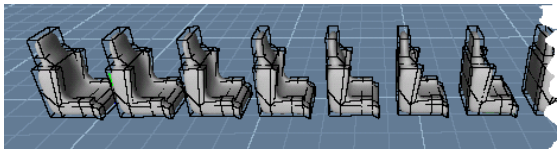
Recording on

1. Create a chair and make sure it is selected.

2. Switch to the top view and activate macro recording by selecting the Macros/Record pull down menu. This operation clears the previous current macro and enables macro recording.


Duplicate and move the chair

3. Select the Edit/Duplicate pull down menu and use the Move tool to move the duplicated chair to get a second chair.



Duplicate+Move macro executed a couple of times

4. Now reset macro recording by selecting Macro/Record again. Then select Macros/Execute a desired number of times to create the rest of the chairs.

Rotate a view

When macro recording is activated, by default all applied actions are recorded to macros. For example, you can record a macro, which rotates the view window.

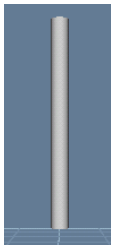
1. Set macro recording on

2. Rotate the view window by holding down the Alt key while dragging with the right mouse button.

3. Clear macro recording. Now hit the key combination 'Ctrl g' to execute the current macro (or select the pull down menu Macros/Execute). The view window is rotated.

Spiral stairs

You can record a macro, which first moves an object, then rotates the view and finally moves the selected object again. Such a macro contains two kinds of actions: those which are applied to the view window and those which are applied to the selected objects.



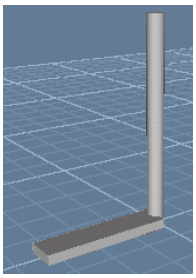
A center pillar for spiral stairs

Sometimes you don't want view specific events to be recorded into macros.

Macro filters prevent actions applied to a certain target, such as the view window, being recorder to a macro.

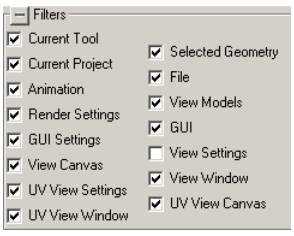
This tutorial demonstrates how to create spiral stairs using macros. The idea is that each step of the stairs can be created by duplicating, rotating and moving the previous step. These three functions can be recorded to a macro. The stairs are then built by executing the macro a desired number of times.

1. Create a cylinder representing the axis of stairs.



The first step created

2. Create one analytic cube representing the first step.

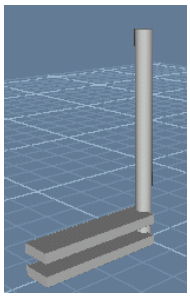


Disable macro components, which affect view's options

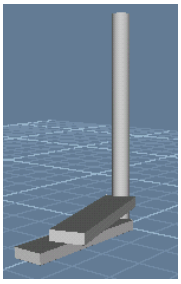
3. Go to the Select window's Macro tab (the third last one) and open the Filters frame. By default, all macro targets are enabled. Reset the View Settings check box to disable events, which change view orientation.

4. Activate Macro Recording. Make sure the step object is selected and apply the Edit/ Duplicate pull down menu. This records the Duplicate operation to the current macro.

5. Switch to the side view. Because macro recording for view window is disabled, view rotation is not recorded into the current macro. Move the duplicated step upwards using the Move tool.



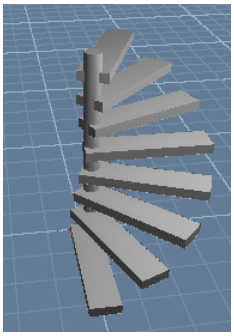
Duplicated step translated upwards



Second step ready

6. Switch to the top view and rotate the duplicated step using the Rotate tool.

We have now created the second step by duplicating, moving and rotating the first one.



Macro executed six times

7. Reset macro recording and make sure the second step is still selected. Select the Macros/Execute pull down menu to create the third step for the stairs. Repeat 'Macro/Execute' a desired number of times to get all steps created.

The stairs are now ready.

Entering points from a macro

In this example, we create a simple macro, which just enters a number of points through the view window. Before you try this example, make sure the View Settings macro target is activated in the select window's Macro/Filters controls.

1. Set macro recording on

2. Click the left mouse button several times in the view window. Nothing happens because you haven't activated any creation tool. However, these mouse clicks are still recorded to the current macro.

3. Disable macro recording. You have now recorded a macro, which simply feeds a number of points to the current tool, whatever it is.

4. Click the NURBS/Curve tool and hit the key 'Ctrl g'. A smooth NURBS curve appears to the view window. Click Accept to finish the tool.

5. Activate the NURBS/Curve tool again. This time set the order to Polygonal. Then execute the current macro and click Accept. A polygonal NURBS curve is created through the recorded points.

Activating tools from a macro

You can record a macro, which only activates a tool, but lets you interactively define the necessary points for the tool. The purpose of such a macro is to save the work of selecting the suitable tool options.

1. Set macro recording on.

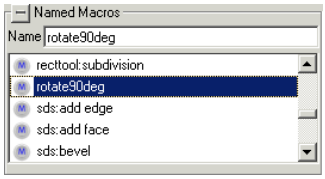
2. Click the desired tool and set suitable tool options through the control bar.

3. Disable macro recording and cancel the tool. Hit the keys 'Ctrl g' to activate the tool with the specified options.

Creating named macros

The latest recorded macro is named as 'current' by default. Whenever you record a new macro, the old current macro is destroyed.

If you want to keep a macro more permanently, you have to rename it. This can be done as follows:



Named macros

1. Record a macro as described in the previous examples.

2. Go to the select window's macro tab and open the Named Macros frame.

3. Activate the Name field, enter a new appropriate name and hit Enter.

Executing a named macro

You can execute any named macro by dragging & dropping it into a view window.

If you want to repeat a macro multiple times, just select it from the named macro list, enter a suitable count to the Repetitions field and click the Apply button.



Repeat a macro 5 times

Particles

Particles are objects with a very simple geometric description. Due to their simplicity, they are memory efficient and a large number of them can be created.

Phenomena such as water drops, fur, fog and fire, can be simulated using particles. Particles can be rendered in a variety of ways: as small dust particles, lens flares or foggy spheres, to name a few examples.

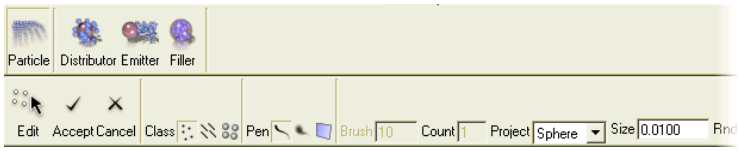
Despite their simplicity, the rendering engine of Realsoft 3D can make particles look very interesting. For example, particles can be textured just like other geometric objects.

The Particle tool allows you to create particles interactively.

To create a flock of particles:

1. Activate the particle tool.

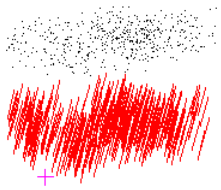
When you activate the tool, the Control Bar shows you a number of particle specific options.



Particle tool activated. The Control Bar shows you creation specific options.

2. Specify desired options from the Control Bar. To get started, just use the default values. Then create particles interactively through the view window by holding down the left mouse button while moving the mouse.

When done, accept the tool by clicking the right mouse button and selecting Accept from the opened pop-up menu.



1D and 2D particles

You can also airbrush particles on the surface of the selected objects. To do this, set the Surface option of the particle tool. Orientation of 2D and 3D particles will then be determined by the orientation of the underlying surface. This option allows you to create easily fur and hair for objects.

Below is a description of particle tool specific options.



Particle tool options

Class

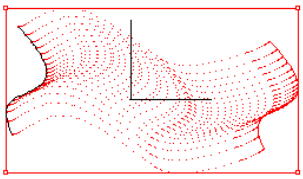
The type of the created particles. Currently three kinds of particles can be created:

- **1D**. 1D particle is defined by a single point. Rendering engine can render 1D particles as points, circles, stars etc. 1D particles look the same from all directions.
- **2D**. A 2D particle consists of two points. So, it has properties such as direction and length. Real world objects like water drops, which have a symmetry axis, can be rendered using 2D particles. Also, polarity can be simulated (particles can have positive and negative ends).
- **3D**. The geometry of 3D particles is represented as a coordinate system. In other words, 3D particles can be scaled, rotated and skewed. They can be used for rendering objects, which look different from all the three main directions. Please note that the current scan line shaders do not render 3D particles visible.

Pen

Currently there are three pens available:

- **Pencil**. This creates one particle for each mouse movement.
- **Brush**. The number of particles created for each mouse movement can be defined through the Count field.
- **Object**. This creates particles along a selected NURBS curve. The number of particles generated for each mouse move event can be defined using the Count control.



A NURBS curve used as a pen for creating 1D particles

Brush

This field controls the radius of the airbrush particle tool. The size is measured in pixels.

Count

Defines the number of particles created per each mouse movement when using the brush and selected object pens.

Project

This option controls the depth coordinate of particles added by the brush pen. Three choices are available:

- Disc: Particles are created into the input plane. The result is a flat particle cloud.
- Sphere: Each mouse movement adds a new spherical cloud of particles around the mouse position defined by the input plane. The diameter of the particle cloud is defined by the Brush gadget.
- Surface: Particles are added to the surface of the selected objects. The surface also defines the orientation of the particles.

Size Defines the size of created particle items.

Rnd Size

Randomizes the size of created particle items. 0=constant size, 1=fully randomized size.

Rnd Dir

Randomizes the direction of created particle items. 0=fixed orientation, 1=quite random orientation dominated by original direction, 10=totally random orientation.

Modifying particles

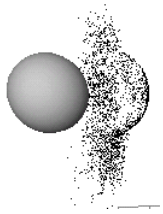
Particles can be modified just like any regular objects, such as spheres and NURBS curves.

For example, you can modify the created particle flock by using the move tool with Collision Deformation option set:

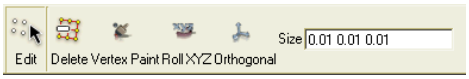
1. Select the particle object and activate the move tool with the Collision deformation option.



2. Move the particle object against the sphere.



When you select a particle object, the control bar automatically shows you particle specific tools.



Particle tools

You can also enter the edit mode to single point edit the particles.

You can use any of the available tools, such as move, rotate or bend, to modify the selected particles. You can duplicate points simply by dragging them while holding down the copy modifier key 'Ctrl'.



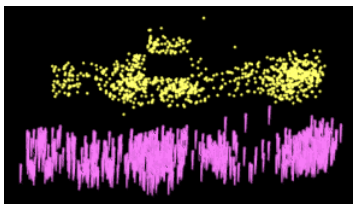
Some of the particles selected in edit mode

Properties of particles

All common object properties, such as name, color or mass, can be modified through the property window.

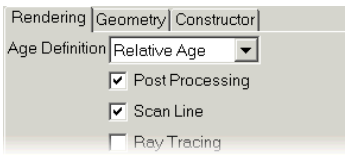
The color of a particle object can be defined in the usual way using the property window's Col tab.

The default scanline shading of particles does not look especially pretty.



1D and 2D particles rendered

Particle specific options can be controlled through the Property Window's Spec tab. It includes three sub tabs, which are described below.



Particle specific properties

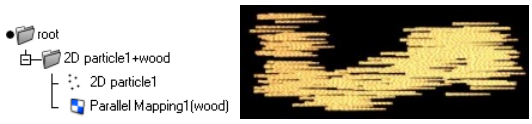
Rendering tab

Particles can be rendered using *scanline*, *post processing* and *ray tracing* techniques. These are not exclusive: you can render a scanline particle and add a post effect on top of it. For more information about how to define visual appearance of particles, see the Post processing chapter.

Scan line rendering

The color, optical and other surface properties of scanline rendered particles can be defined in the same way as those of any regular objects - by using the material system.

For example, you can texture map scanline particles using the wood material of the default material library.



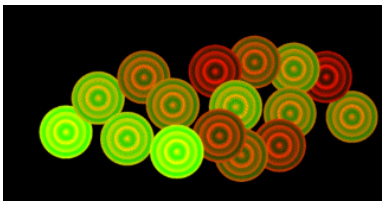
Texture mapped 2D particles. Wooden particles consist of two sub objects: 2D particle and wood.

To texture particles:

1. Create a set of particles. Select the particle object.

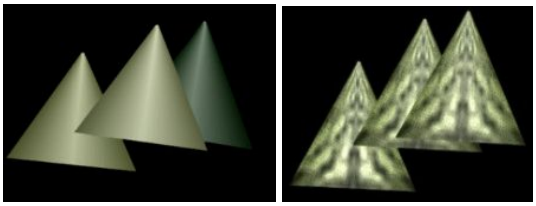
2. Drag&drop a material from the Materials tab of the Select window into the view window. This creates the object hierarchy shown in the example image.

Particles have a well defined surface parameter space, and therefore you can use UV mapping to map materials to particles. The UV space of 1D particles is two dimensional. The 'u' coordinate is defined in the radius direction. It is zero in the center of a 1D particle and runs to one with the radius. The 'v' coordinate increases with the order number of individual particles: the first created particle has the v coordinate value 0, and the last particle has v equal to one.



UV mapped particles. Color changes by the 'u' coordinate.

The UV space of 2D particles is three dimensional. 'U' runs along the length of a particle. 'V' runs perpendicular to the direction of the particle (from the axis towards the edge). The 'W' coordinate runs with the order number of the particle. Using the UV coordinates, you can define materials, which for example, make 2D particles look like 3D cylinders, cones, or hairs. And of course, 2D particles can also be bump mapped, clip mapped, they can be transparent or reflect light.



2D particles whose color fades by v coordinate making them look like 3D cones, and bump mapped 2D particles

Scan line rendered 1D and 2D particles are visible in ray tracing. For example, other objects can cast shadows on particles.

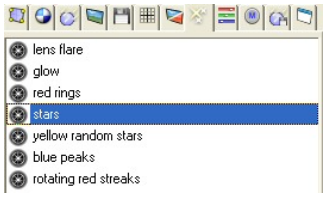
The only drawback with scan line rendered particles is that they are quite memory consuming. However, a high render box count in rendering decreases memory consumption. Summary: defining material properties for scanline particles does not differ much from defining material properties for any regular geometric object such as a sphere or a NURBS mesh.

Post Processing

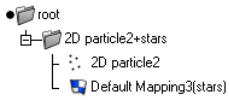
In addition to scan line rendering, also post processing can be used for rendering particles. To do this:

1. Select the desired particle object.

2. Drag&drop a post particle effect from the select window into the view window.



Drag&drop the desired particle effect into the view window to map it to the selected particle objects

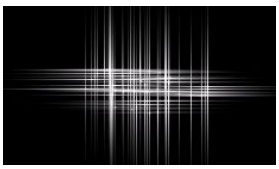


Star filtered particles

As expected, this creates the object hierarchy shown in the adjacent image.

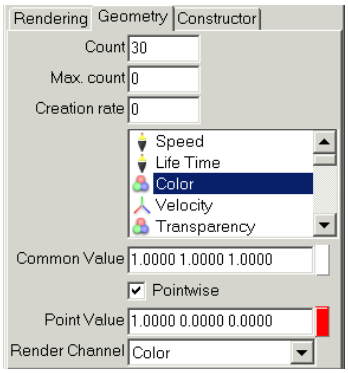
Rendering this shows a number of stars.

Note that several rendering methods (scanline & post processing) can be used simultaneously. For example, you can use the scan line particles to render hairs and then use post processing to make the hairs glow.



A star effect mapped to 1D particles and rendered

Geometry tab



Geometric particle properties

The Geometry tab controls particle specific properties such as size and length. It includes the following fields:

Count

Controls the number of particles. If you increase the count, position and other properties of added particles are computed by interpolating the existing particles.

Max Count

Limits the maximal number of particles. The value is used, when a particle constructor is enabled, making the count vary over time.

Creation Rate

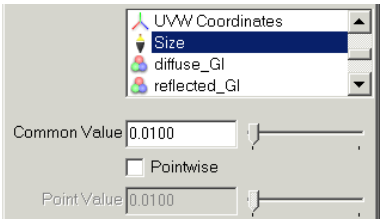
Defines how rapidly particle count increases, when a constructor is enabled. The value defines the total number of added particles during the whole animation. Note that the rate value can be animated, making the density of particle flow variable. A negative value stops particle recreation (particle count starts decreasing, when existing particles die).

Properties

This list view allows you to select and define particle specific properties, such as life time and speed.

The rest of the fields in this tab are used for defining initial values and other properties for the selected attribute.

Common Value



Uniform size defined for 1D particles

This gadget shows the common 'per object' value of the selected property.

For example, to modify the size of 1D particles:

1. Select the Size property from the Attribute list.
2. Enter a desired radius to the Common Value field.

Pointwise

Each property can have a uniform value or it can be defined per particle. Uniform means that the all particles in the object share the defined property.

By checking the Pointwise box, you can assign an individual value for each particle. For example, to modify the size of particles:



2D particles here and there

1. Select the Size property from the Attribute list.
2. Enter a suitable base value to the Common Value field. This value will initialize per point values.
3. Check the Pointwise option.
4. Set the particle object to edit state and select some particles from the view window.
5. Enter a new radius to the Point Value field.

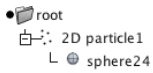
Render Channel

This gadget selects a channel for holding the value of a pointwise property in rendering. For example, you can define that a dynamic float channel Age holds the value of the Life Time attribute of particles. Then you can make a simple VSL material, which reduces the color of particles by the Age channel.

Constructor tab

Constructor

This control allows you to control the shape of the particle flock using other geometric objects. Distribute over surface option distributes the particles over the surface of the sub objects. To use this feature:



2D particle object contains a sphere

1. Create some 2D particles and create a sphere inside the particle object.

2. Select the particle object and open the property window. Go to the Spec/Geometry tab and activate the Distribute over surface constructor.

3. Modify the sphere. Particles remain distributed over the sphere.


2D particles distributed over a sphere

Animating particles

Particles can be animated in the same way as any geometric objects.

Key framing and Morphing Particles

Particles can be key framed the same way as other geometric object. For example, to morph particles, simply turn on animation recording and single point edit the particles.

Path Animations

Particle objects also support path animations.

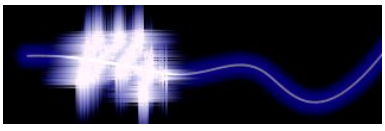
For example, you can use pointwise mapping to map particles to a path to achieve animated path deformations:

1. Create a set of particles with the Particle tool.

2. Create a NURBS curve.

3. Select the particle object and then the curve object and apply the pull down menu Animation/Path/Pointwise.

4. Select the particle object and drag&drop a desired post particle effect (such as the star effect) into the view window. You get a bunch of stars traveling along the path.


Particles following a path

Simulations

Particles are often animated using the simulation system. For example, you can use a number of 1D particles as meteorites acting under the influence of a gravity field caused by a planet.

To do this:

1. Create a number of 1D particles representing meteorites.

2. Create a sphere representing a planet.

3. Multi select the sphere and the particles and apply the tools Animation/Simulations/Gravity and Animation/Simulations/Collision Detection. Open the Property Window for the sphere, go to the simulation tab, and change Gravity and Collision Detection to Cause.

4. Select the particles and apply the tool Animation/Physical Properties/Warm butter.

5. Set the frame count to 500 and play the animation.



Note

Using the simulation system to control particles does not differ from using the simulation system to control any regular object. However, note that particles don't have volume, and therefore two particles cannot collide. Particles can collide with true 3D objects, such as spheres, without any problems.

Creating Leaves Using 3D Particles

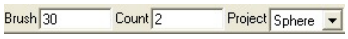
This tutorial shows how to simulate tree or plant leaves using textured 3D particles.

Beginner/intermediate level.

Click on the Particle tab and select the Particle tool.



Particle tool selected, set to create 3D particles using the airbrush pen

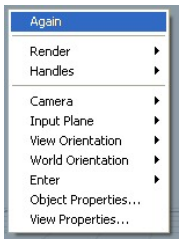


Spherical brush projection selected

Select the 3D particles and select the Airbrush pen. This allows you to literally spray particles around. Next set the Brush radius to 30, Count to 2 and Project to Sphere.

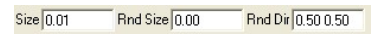
These settings allow you to spray just a few 3D particles with every mouse click. This also causes the particles to be sprayed around using a spherical brush, meaning that the particles are not all in the same 2D plane, but spread in all 3 dimensions, only limited by the size of the brush.

To see how this spherical brush works, just click and hold the right mouse button for about 10 seconds and then accept the tool. In your view window the particles will form a 3D sphere about the size of your brush. Press Ctrl + Z to *undo* last action. Then, in the view window, press the right mouse button and select Again from the pop-up menu. This restarts the particle tool with the same settings as you had the last time you used it.



Select Again to restart the tool with the same settings as last time

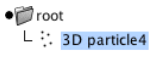
The particle tool has some additional controls for size and direction. Set Size to 0.1, RndSize (Random Size) to zero and RndDir (Random Direction) to 0.5.



Settings for Size, Random Size and Random Direction

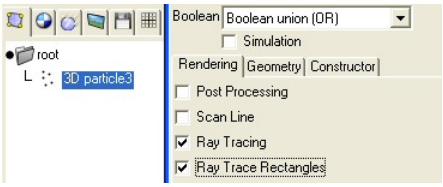
If RndSize is set to zero, all particles will have the same size. If RndSize is set to a higher value, more and more particles will have a different size. The same goes for RndDir: if set to zero, all particles will have the same orientation towards the camera; change that value and their orientation will change randomly.

All settings are OK now. So, spray just a few particles around by clicking and releasing the left mouse button at various locations in the view window. If you click and hold the left mouse button, more particles are added to the scene. When done, accept the tool. A 3D particle object is now added to the hierarchy in the Select Window.



From spheres to rectangles

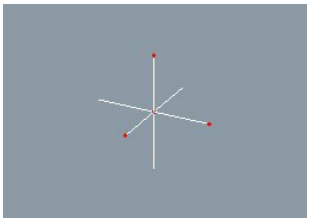
If we now render the scene, all 3D particles will render as spheres. This is the default setting. Because we want to make the particles look like leaves, we need the shape to change from spheres into rectangles. This is simply done by changing the properties of the 3D particle object. Double click on the 3D particle object in the Select Window to open it's properties. Go to the Spec tab of the property window. On the Rendering tab make sure that Post Processing and Scan Line are unchecked and then check Ray Tracing and then Ray Trace Rectangles.



Ray Tracing and Ray Trace Rectangles options set on the Rendering tab of the property window

The Ray Trace Rectangle option ensures that all particles are rendered as rectangles, which can be textured. In our case, we can texture them now to look like leaves. And because we set the RndDir option as well, not all rectangles (leaves) will face the camera, which enhances the effect we need.

When the Ray Trace Rectangles option is set, all 3D particles are rendered as equally sized squares. This is of course due to the settings we used when creating them (RndSize = 0). It is also possible to edit the size and shape of each individual particle inside the 3D particle object after we created them. Attributes like size, shape, position and orientation of the individual particles can even be keyframed for animation.



A single 3D particle in edit mode

To change the size of a single particle, just select the particle object in the Select Window and press the Space bar on your keyboard to go in edit mode. The 3D particles have 3 points that can be edited and one center point that can be used to move the particle around.

Modifying a the shape of a particle is simple: just drag the red dots on each axis until the particle reaches desired shape. To move a particle, just drag the red dot in the center of the particle to a new location in your view window. If you wish to move several individual particles, hold the Shift key and click at the center dot of each particle you wish to move. You can also use the Drag Select method to multi select particles. To modify all selected particles simultaneously, use the Drag Select method to multi select particles and then apply the Noise tool, which is located in the Transformation toolbar .

Creating the Leaf material

Create a new VSL material, call it Leaf1. Check the Preview box. We need a leaf texture and an alpha map of the same leaf. So why not use these two images?



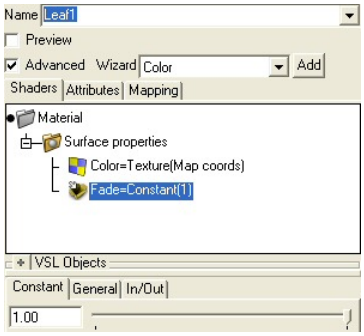
The color map (D_longleaf.gif) and the alpha map (A_longleaf.gif) of our leaves (Thanks Bernie den Hertog!)

Just save these two images to your favorite directory and use them in this tutorial.

Use the texture wizard and select the colored leaf texture (D_Longleaf.gif). Leave the Tile X,Y and Grade X,Y boxes empty. Check the Advanced checkbox to get access to the VSL statements.

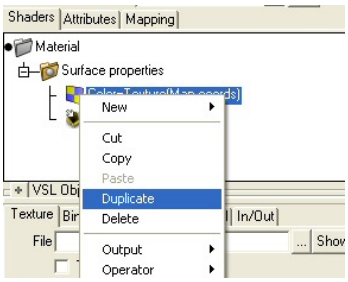
Select a Constant object and drag that into the Surface properties shader. In the lower section of the property window there are three tabs, related to the Constant object. We start with the *third* (most right hand) tab. The third tab is called the In/Out tab. Make sure that Output is set to Surface:Fade.

The second tab is called General. The Operation should be set to '=' (the default value) and do *not* check the other boxes. Finally, go to the first tab (Constant) and set the Constant value to 1. Notice that the preview box turns empty!



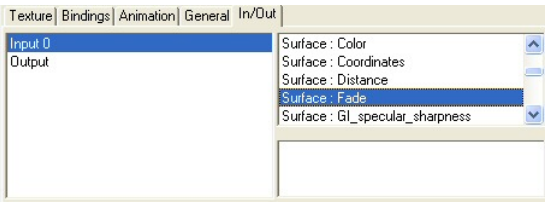
The Constant object added, output set to fade and constant value set to 1

Select the Color=Texture(Map coords) instruction, press the right mouse button and select Duplicate from the pop-up menu. The duplicate instruction will be placed at the bottom of the hierarchy.

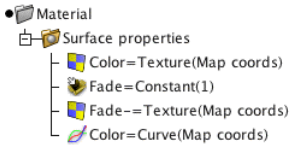


The Color=Texture(MapCoords) instruction will be duplicated

Select the duplicate instruction. In the lower section of the property window there are now five tabs. Go to the Texture tab and select the alpha map image (A_longleaf.gif). Next, click the General tab. Set Operation to '-' (the minus sign) .Finally, go to the In/Out tab. Click on Output and next set the Output to Surface:Fade, simply by selecting it from the list on the right side of the window.



Finally, drop a VSL curve object into the hierarchy, which gives us color control over the texture. At the In/Out tab of the curve object, set both Input0 and Output to Surface:Color. The shader should now look like in the example image.



The material is completed

And then...

Map the Leaf1 texture to the 3D particles using a *default mapping*. Add a simple infinite rectangle to have a background and add a light source. Press render, and you should get something like this...

Now this scene can be made more interesting, if not all leaves have the same color. This is where the VSL curve object becomes handy. Perform the following steps:

1 Create just a few new particles using similar settings as before and now also add some random size, like 0.3.

2. Duplicate the Leaf1 material, call it Leaf2



The first render: leaves-textured particles!

3. Modify the color of the texture in Leaf2, using the VSL curve. This is done as follows. Make sure the material Preview box is checked. In the VSL property window, select the Color=Curve(Color) instruction. In the lower section of the window a graph appears, showing a red line. Click in the middle of the red line. This creates a knot point. Drag the knot point slightly diagonally upwards. Notice the color change in the preview box. When happy with the new color, close the property window.

4. Render again and it already looks more interesting!

Another example

In this part of the tutorial we put our knowledge to use and show some more of the shear power of 3D particles!

First create an analytic cube, which will act as a brick wall. Then create a brick texture and map that to the cube (wall) using a parallel projection. Don't forget to enable the Roll option in the parallel mapping object. You get something like this:



Nice brick wall eh?

Then it's time to grow some plants on that wall. Again we will use our 3D particles with the same Leaf1 material applied to it. Only this time we will use a different projection when creating the particles. This allows us to let the plants follow the structure of the wall. Activate the particle tool again, select 3D particles and the airbrush pen, just like before, only this time with the following settings:



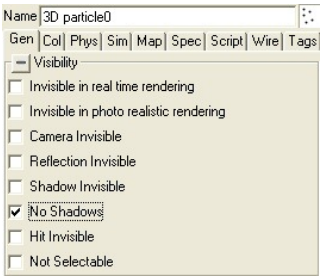
Settings for the 3D particles. Notice the Surface projection and the size of the particles

Make sure the cube (wall) is selected in the Select window. Now spray just a few particles at a time right on the top left corner of the wall and make your way down. Notice that there will be no particles added to the scene when you do not spray on the wall. This is because we set the projection method to Surface and we selected the wall as target.



Leaves grow over the wall

Then open the properties of the 3D particle object, set the Ray Tracing and Ray Trace Rectangle options at the Spec tab. Apply a *default mapping* of the Leaf1 material. Then repeat the particle creation process with the same settings and then apply the Leaf2 material and render.



No Shadows option set, to increase render speed!

When using many textured particles, you may find yourself waiting for your render to finish. Setting the No Shadows option in the property window of the 3D particle object could be a real time saver!

Camera

The rendering engine needs a camera object to define position of the camera, viewing angle and focal length, among other things.

A view window contains an internal camera, which can be controlled interactively with several view navigation functions available through the keyboard, view property window and the view control bar. In addition to this, the user can create any number of camera objects into the hierarchy of geometric objects. View windows can fetch their orientation and other properties from these camera objects.

This tutorial examines how various camera properties can be managed.

Creating a camera

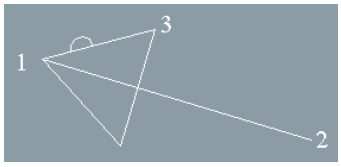
1. Activate the Camera tool from the Creation tab of the Toolbar.



The Camera tool selected

2. Define the camera geometry by entering three points through the view window. These three points define position, aim point and viewing angle for the camera.

You have now created a camera object. You can modify the camera just like any other geometric object using the modify tools (Move, Rotate etc.) and the property window.



Enter three points in a view window to define the position, aim point and viewing angle for the camera

View's camera

A view window uses an internal camera object to define its orientation. When you rotate, pan and zoom the view, you actually modify the position, direction and distance of this internal camera object.

Instead of defining geometry for a camera object under construction by clicking three points, you can also use geometry from a view's internal camera. Using this feature, you can easily save the current state of the view camera to a camera object and restore it whenever needed.

To create a camera object which matches the current orientation of a view window:

1. Activate the camera tool.

2. Accept the tool immediately without entering any points. The created camera matches exactly the orientation of the current view window. (Including orthographic views, i.e. non perspective views!)

Working with multiple cameras

You can add any number of cameras to your project using the two creation methods described above.

To fetch the viewing angle and other settings of a camera object into a view window, simply drag & drop the object from select window to the view window.

If you have included multiple cameras to a project and want to start file rendering or animation preview, you have to first define which camera is the Current camera. By animating the Current camera option, you can also jump from one camera to another during animation.

To make a camera the current camera:

1. Select the camera in the Select window

2. Click the Current icon in the tool control bar

The camera backdrop tool

You can define backdrop images using the Backdrop tool. A backdrop image is a plane defined in a camera's space.

Because a backdrop image is a true 3D surface, other objects can reflect it, cast shadows on it and so on. For example, you can create a magnifying glass, which really magnifies the backdrop image.

Any number of backdrop images can be created per camera. Only the backdrop images of the Current camera are active in file and view rendering.

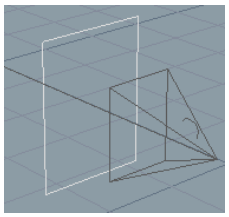
The Backdrop tool allows you to create most commonly needed backdrop effects easily.

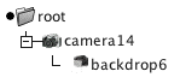
To define a constant background color:

1. Click on a camera in the Select window. This makes the Backdrop icon show in the control bar. Now click on the Backdrop icon and define the desired color to the Color field of the tool control bar



Backdrop tool activated, constant white color defined





A backdrop object shown in the view and the select windows

2. Select Accept on the desired view window.

This automatically creates a new camera object with a backdrop object. If a camera object was selected when applying the tool, the backdrop object is inserted into it (a new camera is not created).

To preview the backdrop in a view window, simply render the view. If the scene contains multiple cameras, make the camera Current first.

Backdrop images are represented as planes in view windows. You can move a backdrop object like any other geometric object. However, you can't shear or scale it. The backdrop always remains perpendicular to the focal axis of the camera.

Backdrop tool sets up a backdrop image, which always renders with the specified color. Light sources don't affect the shading of the backdrop image in any way. This result is achieved by setting the actual diffuse color of the backdrop to black and assigning the desired RGB value to the Illumination channel. To change the unshaded constant color of a backdrop:

1. Select the backdrop object.

2. Open the property window and go to the Col tab.

3. Select Illumination from the Attribute list and enter a new color to the value field below the attribute gadget.

Activating a lens flare effect for a camera

To activate a lens flare for a certain camera:

1. Select the camera.

2. Open the property window. Go to the Spec tab. The Lens Flare gadget shows the contents of the current library of post particle effects (you can add new alternatives using the Post Particle Effects tab of the select window). Select a suitable alternative.

To preview the lens flare, drag & drop the camera from the select window into a view window and click Render.



Note

You can render a lens flare in a view window without a camera object. The view property window contains a camera property gadget (for controlling view's internal camera) and you can select the effect directly using it.

Camera lens flare is a camera specific effect. Lens flares are automatically created by the light sources, which are directly visible from the camera. You can add other kinds of lens flares to your scenes using the post processing system of Realsoft 3D. You can, for example,create thousands of particles and render them as lens flares. This approach provides much more control than the 'automatic' camera lens flare feature. Multiple effects per light position can be defined and flares can be added only to certain light sources by placing a flare-mapped particle to the desired positions.

Depth of field

Depth of field is another camera specific effect. A camera object provides you with an intuitive interface to this effect.

To activate depth of field for a camera, just check the Depth of field check box in the property window's Spec tab. The aim point handle of the camera allows you to specify the distance at which scene renders sharp. By animating the position of the aim point, you can easily animate depth of field. The amount of depth of field depends on the focal length of the camera and F-stop value just like in real world cameras.



Note

You can also control the depth of field effect without a camera object. The module, which renders the effect, is included in the Default Effects configuration, which is available in the Post Image Effects tab of the select window. If you select the included depth of field effect and clear the Camera defined option using the property window, you can control its parameters independently of camera objects. See the Post Processing chapter for more information.

Animating cameras

Camera objects, just like any other geometric object, can be animated using various methods. For example, you can key frame cameras, use gravity to attract them, map them to a motion path and so on.

The key framer allows you to animate the position and orientation of the cameras easily. The program uses an advanced quaternion interpolation system to achieve smooth and intuitive camera rotations.

In many cases, either the camera rotates about its aim point or the aim point is rotated about the camera position. If you want to achieve the former effect, just drag the pivot point of the camera into the aim point. Then rotate the camera using the rotation handles to key frame it.

When using motion paths, banking needs some special attention. The path animation tool provides the most common banking control types. For example, you can select airplane type banking.

Lattice mapping system allows you to set up camera tracking animations easily. For example, you map the aim point of a camera to a moving object to make the camera follow the object automatically.

Connecting a view and a camera object

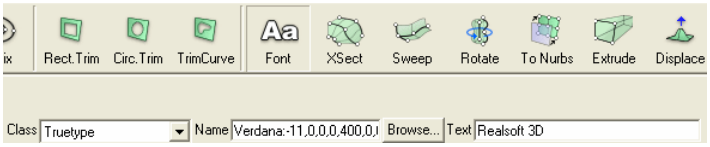
The view window supports a Tracking option. In other words, you can lock a view window to the current camera object. Whenever the camera object is moved or otherwise changed, the view window updates accordingly.

You can activate this feature from the view property window's Camera tab. If you set Tracking = Track animation play, the camera is updated only when animation time is changed.

If you set Tracking = Track always, the view window and the camera become completely synchronized. If you rotate the view window, the camera object is rotated accordingly. If the camera object changes, the view changes with it. Take care using this option when setting keyframes with animation recording ON, make sure to turn this feature OFF when you have finished recording. Otherwise you may ruin your carefully set camera keyframes by changing the view later.

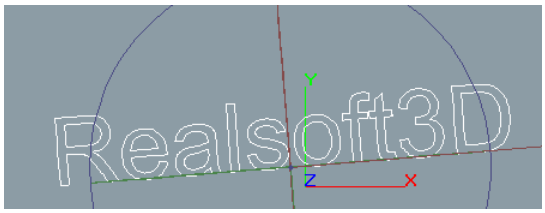
Fonts

The Font tool creates NURBS curves representing a given text string and font type. The generated curves can then be extruded to 3D or used as animation paths, etc.



The Font tool selected

1. Activate the Font tool. It's located in the NURBS tab of the control bar.



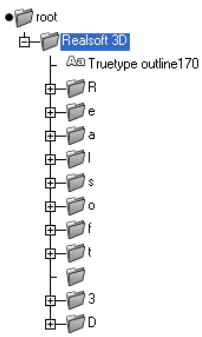
Enter two points through the view window to define a base line

2. Define a TrueType font and some text through the control bar.

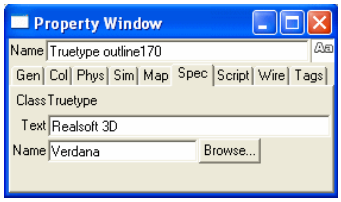
3. Enter two points through a view window to define a base line for the outline curves.

This creates the object hierarchy shown in the example image.

You can modify the generated text any time by selecting the font manager object (such as 'truetype outline69') and editing its text property using the property window.



Font construction consists of a Font Manager object and font curves



Changing Text of a font manager object

For example, to change the Realsoft 3D text to 'Yes':

1. Double click the 'Truetype outline69' object in the select window. The property window is opened, or simply activated if already open.

2. Go to the Spec tab and enter a new string to the Text field.

The font manager object updates the curves to reflect the new string.

To extrude the NURBS curves, simply select all levels below the font manager containing the single letter curves and select NURBS/Extrude from the main toolbar.

Rotator Object

The rotator object creates a number of new objects by copying, rotating and translating its sample objects. For example, you can create the balls of a ball bearing using the rotator object.

To create balls for a ball bearing:

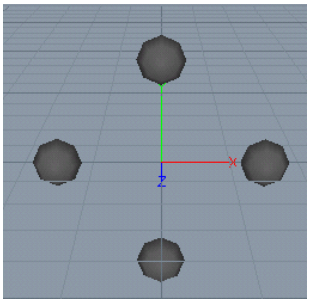
1. Create one analytic sphere

2. Select the sphere and activate the Rotator tool (it's in the Modifier tab). The tool control bar now shows you a number of rotator tool specific options.



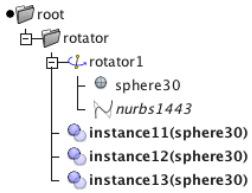
Rotator tool activated

3. Enter a point through the view window to define the center of rotation. This accepts the tool. You now have four spheres rotated about the defined center point.



Four spheres created by rotator tool

In the select window, you can now see a rotator object and three instances. If you click the '+' handle of the Rotator object, you will find the original sphere and a NURBS axis object inside it. The rotator object duplicates the sphere about the created axis object.



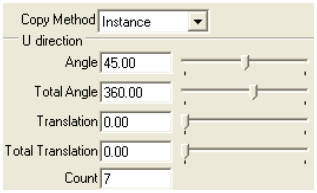
A sphere is instanced and rotated three times by a rotator object

Modifying rotator objects

You can change the number of object copies created by a rotator object (and other options as well) afterwards. For example, to change the number of spheres to 8:

4. Select the rotator object and open the property window.

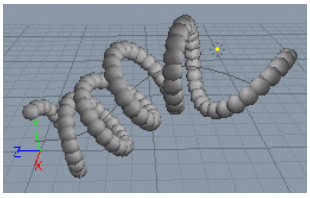
5. Go into the Spec tab and set the U Direction/Angle field to 45 degrees. Then set the Count field to 7. You now have eight spheres (the original sphere + seven instances).



Changing the U direction angle and number of copies

In addition to rotation, the rotator object can translate objects along the defined axis. Value 1 translates spheres by the full length of the axis. You can also replace the straight NURBS line with any curved object to get spheres translated and rotated along a curved path.

If you only define 'translation' and set rotation to zero, the rotator object can be used as an alignment tool.



Translation and rotation along a curve

The rotator accepts also a two-dimensional object, such as a rectangle, as an axis. In this case you can control the rotation/translation in two directions. For example, you can create arrays of objects using two perpendicular translations.

Animating a rotator object

Rotator objects can be animated the usual way: just set animation recording on and modify the rotation options, such as the number of copies, to key frame them. You can also animate the axis and sample objects freely.

Tube Object

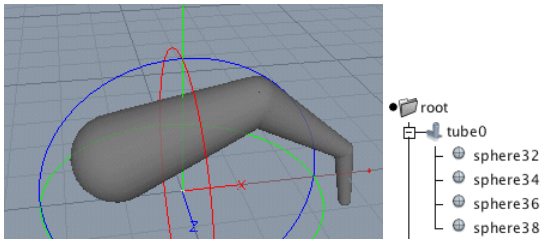
The tube object connects analytic spheres using cylinders and cones.

To create a tube object:

1. Select the New/Tube from the pop-up menu of the select window. This creates a level object.

2. Create the desired number of analytic spheres inside the tube object. The object hierarchy in the select window should look as in the picture below.

The spheres you dropped inside the tube level object are instantly connected with cylinders and/or cones.



A tube connects four spheres with a varying diameter

Editing tubes

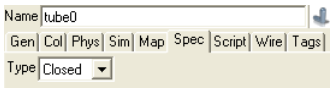
You can freely modify the position and size of spheres inside a tube object. The shape of the tube is updated instantly.

However, you should not stretch the spheres. The reason for this is that it is not possible to connect ellipsoids using cones and cylinders. If you need to create a tube with an arbitrary shape and cross-section, use NURBS or subdivision surface tools instead.

The property window can be used for selecting the desired tube type. For example, to create a closed tube:

1. Select the tube object

2. In the property window/spec tab, set Type to Closed.

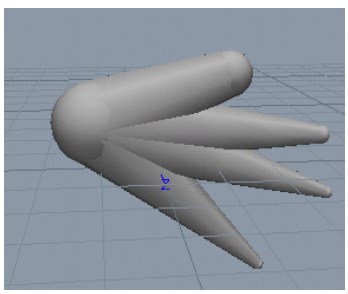


Selecting Closed tube

This connects the first and the last spheres so that the tube becomes a closed loop.

The Star type tube connects the first sphere with the other spheres creating a star shaped tube system.

Tutorial project: 'tutorprojects\modeling\tube\star'



The Star option connects five spheres

Animating tubes

You can animate the shape of the tube by animating its sub objects.

For example, you can create a fancy animation by using collision detection and gravity to control the sub objects as follows:

1. Create a tube object consisting of a set of spheres.

2. Select the spheres. Then select the menu Animation/Simulations/Gravity from the menu bar.

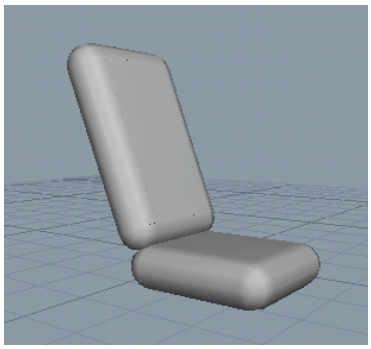
3. Select the menu Animation/Simulations/Collision Detection.

4. Play the animation.

Tutorial project: 'tutorprojects\modeling\tube\tube through colliding spheres'

Instances

Consider a situation where you need to model hundreds of chairs for a cinema theatre. The chairs all look the same, only their position and orientation vary.



A sample chair

Instances allow you to manage this kind of situations efficiently. An instanced object duplicates its source object. When the source object is modified, all instances referring to it will change accordingly. Instances can be transformed or deformed using any available modify function including single point editing. Instances can be instanced, instances of instances can be instanced, etc.

For example:

1. Model a chair of a movie theatre.

2. Make sure that the chair is selected. Then select the pull-down menu Edit/Instance.

Modifying Instance Objects

You can modify an instance object using all common tools, such as delete, duplicate, move, rotate and scale:

3. Select the instance of the chair and activate the Move tool.

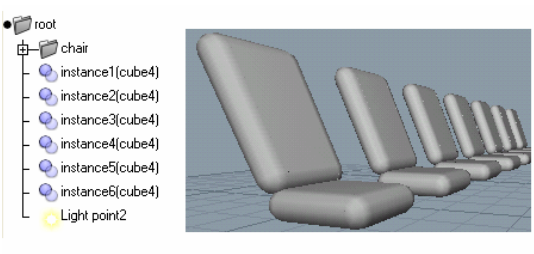
4. Move the instance to the right side of the original chair.

We can use the macro system to achieve more instances quickly:

5. Select Macros/Record from the pull-down menu.

6. Select the instance object. Select Edit/Duplicate. Then move the duplicated instance to the right.

7. Select Macros/Record again from the pull-down menu to switch macro recording off. Go to the Macros tab of the select window. Enter a suitable value to the Repetitions field and click Apply.



A row of chairs

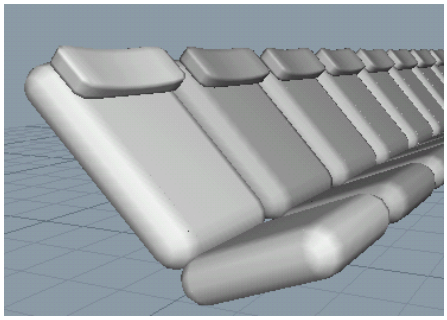
Modifying the Source Object of Instances

Let's improve the chairs. Because of instancing, only one chair needs improvements:

8. Add a neck support to the sub hierarchy of the original chair.

9. Select all sub objects of the original chair and rotate them.

All instances change to reflect these modifications.



More comfortable chairs

Properties of instances

- Instances can be freely animated.
- Levels with many sub objects can be instanced.
- Instances spare RAM memory used for storing object geometry and wire frame representations. Instances also consume less disk space than 'real' geometric objects when saved to a file.
- When the Render as Instance option is unset, the photorealistic render engine converts the instance to an actual geometry (for speed reasons). In this case, instances do not decrease rendering time memory usage. If Render as Instance is active, raytrace rendering becomes very memory efficient but usually a bit slower.
- Instance property gadget has a control which specifies how transforming the source object or its parent levels affects the instance. By default, the Ignore Matrix option is set, meaning that you can move, rotate and scale the source object independently of its instances. To modify both the source object and the instances, you have to make the modification to the sub hierachy of the source object (sub objects or geometry points in edit state). If the option is cleared, the instance system takes into account the modifications performed to parent levels. If you modify the parent level of an object, instances referring to that object become modified as well. In many cases this is the expected behavior. However, there is one side effect: if an instance and its source object are moved into a new hierarchy level, the transformation history defined by the parent levels may change. Therefore, the evaluation of modifications performed to the instance may also change. The following example clarifies this: Create a level, and put an object into it. Create an instance of the object to the same hierarchy level and move it 0.1 m to the right. Rotate the level 45 degrees. Both objects rotate as expected. The instance still contains the information 'source moved 0.1 m to the right', but moving happens now in the rotated parent space. Now create a new level and drag & drop both the instance and the source to it. The instance jumps to a new position, because the new parent level is not rotated.

Constraints

Assume that you have to create a sphere whose 'x' coordinate always stays right in the middle of the 'x' coordinates of the two other spheres. However, in 'y' and 'z' directions, you should be able to move the sphere without any restrictions.

Constraint objects allow you to set up this kind of relations between objects. To define the relation outlined above:

1. Create three analytic spheres and name them as 'sphere1' 'sphere2' and 'sphere3'

2. Select the sphere1 and activate the Constraint tool. You can access this tool from the menu (Tools/Modifier/Create Constraint). Click Accept.

3. In the select window, drag and drop sphere2 and sphere3 into the created 'constraintX' folder.

4. Select the constraint object, open the properties window and go to the Spec tab. Set Command Language to JavaScript and enter the following program:

```
var i = 0;
var mid_x = 0.0;

for(i = 0; i < 2; i++)
{
    child = Self.GETSUBBYORDNUM(i);
    if(!child)
        break;
    p = child.GetCenter();
    mid_x = mid_x + p.x;
}

if(i != 0) {
    mid_x = mid_x/i;
}

p=Target.GetCenter();
p.x=mid_x;
Target.SetCenter(p);
```

The constraint defines all attributes of the associated objects in its own attribute space.

Field Evaluator

The field evaluator object provides you with useful information of an object, such as true velocity, spin, acceleration and angular acceleration. The animation chapter presents how to use the field evaluator to create an object whose muscles are deformed by acceleration. This chapter will show you how to evaluate material properties in any point in space and use the evaluated property to control other objects.

Tutorial level: Medium

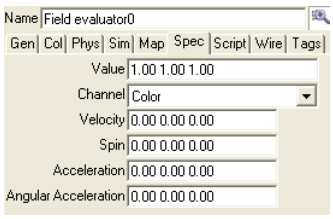
Evaluating color

To find out the color in a certain point in space:

1. Create a field evaluator object by selecting New/Field Evaluator from the pop-up menu of the select window.

2. Move the origin of the field evaluator object to a desired position.

3. Open the property window and in the Spec tab, set the Channel field to Color. You can read the color from the Value field.


Name Field evaluator0
Gen Col Phys Sim Map Spec Script Wire Tags
Value 1.00 1.00 1.00
Channel Color
Velocity 0.00 0.00 0.00
Spin 0.00 0.00 0.00
Acceleration 0.00 0.00 0.00
Angular Acceleration 0.00 0.00 0.00

The color in the specified point is 1.00, 1.00, 1.00

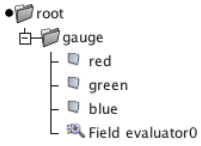
If you want to find out the transparency, just set Channel to Transparency.

This kind of tool is naturally useful for examining object properties. Furthermore, you can use the evaluated information to drive choreographs.

Gauge

Let's animate object lengths based on the color of a given point in space. Three rectangles represent a sort of gauge object. When we move the gauge to a desired position, the lengths of the rectangles immediately show us the color in that point.

There is no such built-in feature in the program. However, we can create it easily using the field evaluator object. The idea is simple: first we model three rectangles, red, green and blue. Then we make their length depend on the evaluated color.



A gauge object consisting of three color indicators and a field evaluator object

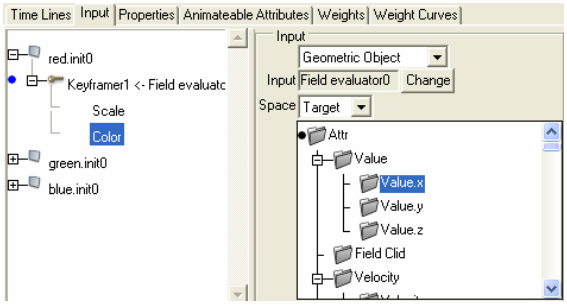
1. Create a level object consisting of three rectangles (red, green and blue). Create also a field evaluator object.

2. Select the field evaluator object and in the property window's Spec tab set the Channel attribute to Color. This way we ask the field evaluator object to evaluate the color property.

3. Animate each rectangle so that they grow longer as the time proceeds. You can also rotate the rectangles so that they behave more like a traditional speed gauge. Or, perhaps you want to animate the color of the rectangles.

The only thing we need to do now is to use the field evaluator to control the choreographs of the rectangles. This can be achieved as follows:

4. Open the choreography window and select the Input tab. Select the first keyframe choreography, which modifies the red rectangle. aDrag the field evaluator object from the select window and drop it over the selected choreography. Then select the Value.x from the input attribute list. Repeat this step for the remaining two choreographs. Then switch to the Time Lines Tab. Select again each choreography and set its Start and End values to 0 and 1, because the RGB values typically range between 0 and 1. The gauge is now ready.

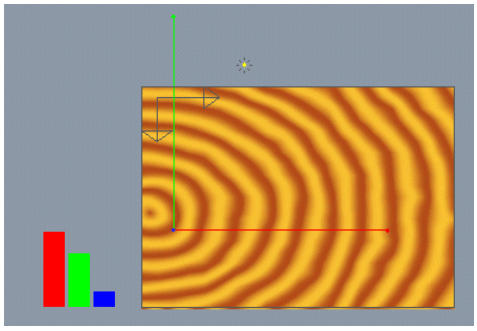


Use the field evaluator's Value components to drive the choreographs

To test it, just create a material mapping object (such as a parallel map) anywhere in space and move the object space origin of the field evaluator to the desired point. The lengths of the three rectangles change immediately to show you the color in that point.

Note: you can freely move the field evaluator in the object hierarchy. For example, to measure the color of a wooden sphere, just drag & drop the field evaluator object into the 'wooden sphere' hierarchy.

Tutorial project: 'tutorprojects\animation\fieldeval\gauge'



The origin of the field evaluator object moved over a textured object. The gauge object shows us the color

Metaball Object

Technically speaking, a metaball object is a special kind of level which interprets its sub objects as a density distribution and computes a connecting boundary surface to a given density iso value. In practice, a metaball 'melts' its sub object together. The melting property makes the metaball object especially suitable for representing liquids.

The first metaball tutorial shows you how to set up a metaball system using a special construction tool.

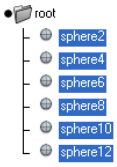


A snowman consisting of six analytic spheres

Tutorial level: Beginner

Example project: 'tutorprojects/modeling/metaball/snowman'

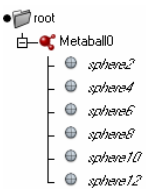
1. Model a snowman using some analytic spheres. As the name might indicate, a metaball object melts only spheres, so do not use other kinds of objects. However, you can freely stretch the spheres if necessary.



Select the spheres and activate the metaball tool

2. Select all spheres. Go to the Creation tab of the toolbar and activate the Metaball tool.

Metaball tool

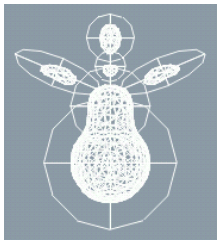


The hierarchy of a metaball system

A new object called 'Metaball' appears in the object hierarchy. Open its sub hierarchy; all the original spheres are placed there.

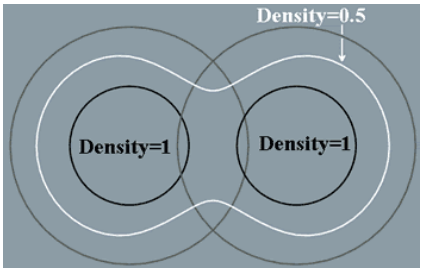
Controlling iso value

The default properties of the 'snowman' metaball are such that the boundary surface shrinks radically from the original spheres.



Shrunk snowman

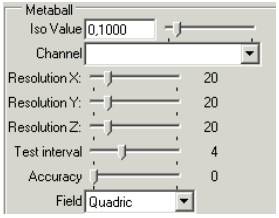
The surface generated by a metaball level follows a given density value in space. Each sphere inside a metaball level adds its contribution to the density field: the density is zero at the outer edge of the sphere, and reaches its maximum value of one at the middle of the sphere radius.



A cross-section of a metaball system consisting of two spheres

3. Select the metaball level. Open the property window and go to the Spec tab.

The Iso Value field defines the density value, which the metaball surface follows. The default Iso Value is 0.5. Change it to a smaller value, say 0.1, and you will see how the snowman rapidly gains weight.



The metaball specific properties

Editing metaballs

The sub objects of a metaball level define its construction history. You can modify the sub objects and add new spheres, and the shape is updated automatically.

4. Select the head sphere of the snowman.

5. Activate the Transformation/Scale tool and expand the sphere. The metaball surface follows the modification.

6. Select the Metaball level. Apply the Make Current menu from the select window's popup.

7. Create two new analytic spheres below the snowman (legs). The spheres are inserted to the current level (= metaball), and the shape changes accordingly.

Alternatively, you can create the spheres at the root level, edit them a bit and finally drag&drop them into the metaball level.

Converting metaballs to another geometry type

Sometimes it may be desirable to convert a shape modeled using metaballs to another form. For example, a metaball surface does not have well defined UV coordinates, which can make applying textures and materials difficult. Fortunately, it is easy to convert a metaball shape to subdivision surface form. The converted shape can be edited further and textured using the SDS tools.

Let's experiment how this happens using the snowman shape:

8. If Metaball is still the current level, select the root level and apply Make Current to it. Then select the metaball level.

9. Go to the SDS tab of the toolbar. Activate the To SDS tool. Make sure that Use Default Resolutions and Subdivision Object options are set. Then click Accept.

10. Note that when the To SDS tool is used, it converts all sub objects to SDS objects as well. We do not need any of these sub objects converted. Hence select all the newly created SDS objects except the last one and delete them. This leaves us with the metaball SDS only.

11. You can now delete the metaball object.

12. Select the created subdivision object and open the property window. Go to the Spec tab. Change Rendering/Type option to Smoothen to Nurbs, and set Quality to the minimum value 1. Close the property window.

13. The remaining step is polygon reduction by converting triangles to rectangles. To do this, select the function Methods/Subdivision Surface/Merge Triangles to Quadrangles.

A subdivision surface that consists of quads usually renders smoother than a triangular SDS mesh, because it is difficult to divide most surface forms to triangles using a symmetric pattern.

Metaball properties

The Spec properties of a metaball include the controls listed below.

Iso Value: The iso value of the density distribution. The generated surface follows this value.

Channel: Defines the channel, which modifies the density distribution. If set, materials that affect the metaball are evaluated to find out the channel value.

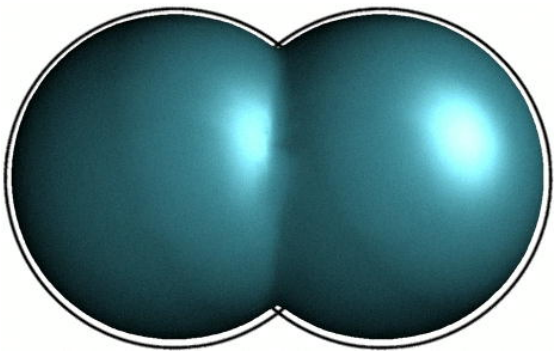
Expansion: Expands (or shrinks) the size of metaballs from their true geometric size.

Resolution X, Y, Z: These gadgets define the overall accuracy of density field evaluation for metaball surface computation. The higher the resolution, the more accurate and smooth the computed surface will be. On the other hand, increased accuracy means longer computation time and higher polygon count on the resulting surface.

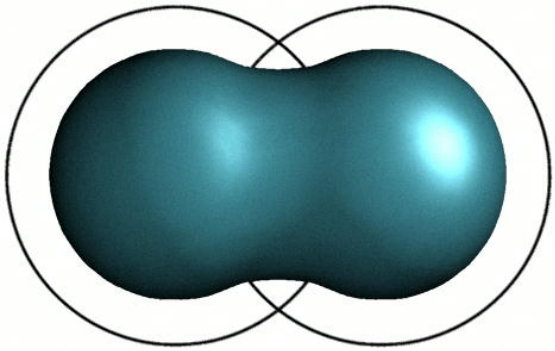
Test interval: An optimization control. If the metaball system does not include small details, or if the Resolution fields are so high that all details will be accurately sampled, the test interval value can be larger than one. Test interval defines how accurately the program examines points defined by the resolution values to find out places where the density really changes. If most of the space is empty, using this control can speed up metaball evaluation dramatically.

Accuracy: Defines how accurately a detected density iso value point is computed. The higher the value, the more exactly the surface follows the given iso value. The default accuracy level zero is sufficient for most situations.

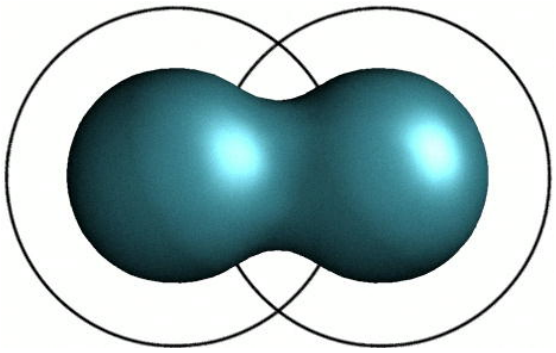
Field: The type of density distribution. Defines how smoothly the density inside a metaball component decreases from one to zero. The following four alternatives are available:



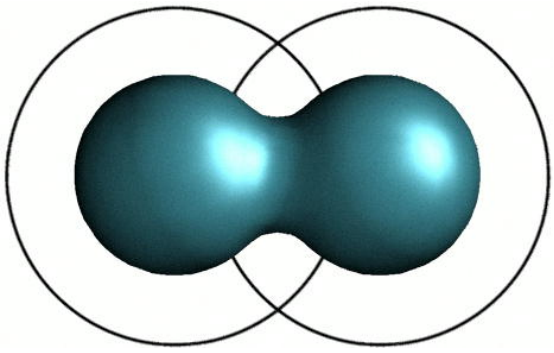
Volume: The density drops rapidly to zero at the boundary.



Linear: The density decreases at a constant rate.



Quadric: The density decreases smoothly to zero.



Cubic: The density decreases extra smoothly to zero.

4th Degree: The density decreases proportionally to the 4th power of the radius. The smoothest way to blend metaballs.

Rendering:

- Triset - the surface is rendered polygonially using the Marching Cubes Algorithm. This is usually the fastest way, but the result shape often suffers from various rendering errors which are typical to polygon models.
- Procedural - the surface is resolved during ray trace rendering without polygonization. This method renders a perfectly smooth surface, when the Accuracy setting is high enough. It is also extremely memory efficient.

Hierarchical: If set, the full sub hierarchy of the metaball level is examined to find metaballs, otherwise only the direct children.

Fluid simulation using metaballs



The next tutorial example, flowing water, combines fluid dynamics and metaballs. The idea is to use a creator object to produce new metaballs (= water drops) during the animation. First read the manual chapter Simulation if you are not yet familiar with the simulation tools of Realsoft 3D.

Tutorial level: Advanced

Example project: 'tutorprojects/modeling/metaball/metafluid'

1. Use the select window's New/Creator to add a creator object to the hierarchy.

2. Create an analytic sphere (radius about 5 cm) and drop it into the creator level. The sphere represents a water drop.

3. Open the property window and go to the Phys tab. Change the mass of the sphere to about 0.15 kilograms.

4. Switch to the Sim tab. Define Gravity = Affected and Fluid Dynamics = Affected for the sphere.

5. Select the creator object. Go to the Spec tab of the property window, and define (approximately) the following values:

- Creation Interval = 0.15 seconds. This value defines how much water comes out from the water pipe. Note: You may have to set the Time System in the Options/Window/Animation tab to 'seconds'.
- Life Time = 3 seconds. The lifetime should be long enough so that water drops have time to fall outside the rendered images.
- Velocity = '0.3 0.05 0'. This value defines how fast water bursts out from the pipe.

Note that these values represent only one possible combination creating a slow water flow. You can experiment and find other values that work as well.

6. Switch to the Phys tab and define a spin value '2 0 0' for the creator object. This changes the direction of water drops slightly during the animation, making it more interesting.

7. Activate the Simulation/Fan tool from the toolbar, and draw a fan object in a view window. Set Turbulence = 1 from the Spec tab of the property window. Then go to the Phys tab and set Density = 10 (dense air will enable slow motion effect). Change Fluid Velocity to 1 m/s.

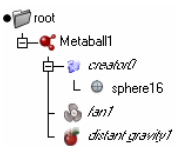
8. Start the Simulation/Gravity tool and draw a gravity line downward in a view window. Using the Spec tab, change Strength of gravity to 0.2.

9. Activate the Simulation option for the parent level of the created objects.

Now it is time to play the animation to check how the water drops move. It is faster to test the simulation part now when the metaball system is not yet included. Rewind the animation and adjust the above mentioned properties if necessary.

10. Select New/Metaball from the select window's popup menu. Drag and drop the creator, the fan and the gravity into its sub hierarchy.

The hierarchy of the scene is now ready. It is shown in the adjacent image.



The structure of the simulation

11. We dropped the simulation components to a new hierarchy level after initial experiments. Therefore, select the metaball level and using the Spec tab of the property window, turn on Simulation again.

12. With the metaball level selected, drag & drop the glass material from the material library to the view window.

Add a nice environment and lighting to the scene before rendering the animation - the kind of environment that gives a realistic look for transparent objects.

Material controlled density field

The regular spherical density field of metaballs can be modified using a material. The following example explains the required steps.

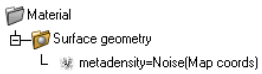
Tutorial level: Medium

Example project: 'tutorprojects/modeling/metaball/asteroid'

1. Create a group of analytic spheres. Select them and apply the tool Creation/Metaball.

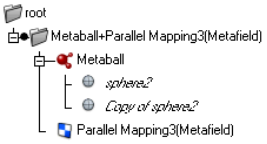
2. Go to the Channels tab of the select window and create a new Float channel. Rename it, for example, as metadensity.

3. Go to the Materials tab of the select window and create a new VSL material called as 'Metafield'. Open the property window and check the Advanced option. First drop a Shader object from the VSL objects frame into the material root folder. Change the type of the shader to Surface Geometry. Then drop a Noise object into the Surface Geometry shader. Change Output of the Noise object to metadensity.



Density field material

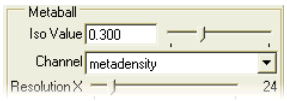
Leave the Base value of the noise to zero and Amplitude to 1, as the noise should vary between 0 and 1 just like the original density field of metaballs. Set the 1D option to obtain faster noise computation (density is a one-dimensional value). Activate the Smooth option, which produces more continuous surface distortion.



Density field mapped to metaballs

4. Select Map/Parallel from the popup menu of the material library. Hold the Shift key down to get an uniform map geometry and draw a faily large parallel map cube over the metaballs in a view window.

5. Go back to the object hierarchy tab of the select window and activate the metaball level, which is now inside a new level. Open the property window. Go to the Spec tab and select metadensity from the Channel gadget. You may also set Rendering to Procedural; that method can properly represent very complex surfaces. However, procedural rendering is often much slower than triangle based rendering. On the other hand, memory consumption is very low and the render quality is much better.



Metadensity channel modifies the density field of a metaball system

6. Adjust the Iso Value so that a suitable amount of the density field generates a surface around it.



Note

The total density field is the product of the spherical density fields and the material defined density field.



An asteroid defined by two metaballs and a density field material

Link Object

A link object points to a geometric object. It can be used where a geometric object would be evaluated.

The first link tutorial shows you how to evaluate a curve object through a link.

Tutorial level: Beginner

Example project: 'tutorprojects/modeling/link/linkpath'



The Link tool selected

1. Create a curve using the NURBS/Curve tool.

2. Select the curve. Go to the Creation tab of the toolbar and apply the Link tool. new object called 'link (nurbs3)' appears in the object hierarchy.

3. Create an analytic sphere near the first point of the curve.

4. Select the sphere and the link object in that order.

5. Go to the Lattice Mapping tab of the toolbar and activate the Path tool.

Click the Accept button.

6. Click the Play button on the Animation window to see how the sphere follows the curve.

Using a link object with particles

Tutorial level: Beginner/Intermediate

Example project: 'tutorprojects/modeling/link/linkparticle'

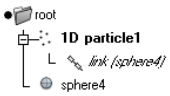
1. Go to the Creation tab and activate the Particle tool. Create a cloud of 1D particles and accept the tool.

2. Select the particle object. Open the Property window. Select the Spec tab, then the Constructor sub tab. Set the Constructor gadget to Distribute over Surface.

3. Create an analytic sphere.

4. Select the sphere. Apply the Link tool.

5. Drag & drop the link object inside the particle level object. The object hierarchy should be as in the adjacent image.



The object hierarchy

The particles are now distributed over the sphere's surface.

Using a link object with fluid dynamics

Tutorial level: Intermediate

Example project: 'tutorprojects/modeling/link/linkfan'

1. Go to the Creation tab and activate the Particle tool. Create a cloud of 1D particles and accept the tool.

2. Select the particle object. Open the Property window. Select the Spec tab, then the Constructor sub tab. Set the Constructor gadget to Emitter.

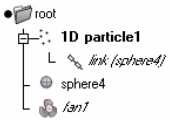
3. Select the Phys tab and set the Mass gadget (inside the General frame) to 0.01.

4. Select the Sim tab and set the Fluid Dynamics gadget to Affected.

5. Create an analytic sphere.

6. Select the sphere. Apply the Link tool.

7. Drag & drop the link object inside the particle level object.



The object hierarchy

8. Create a Fan object pointing at the sphere (The Fan tool is in the Simulation tab of the toolbar). The object hierarchy is shown in the adjacent image.

9. Click the Play button on the Animation window to see how the particles are emitted from the sphere and blown away by the fan.



Chapter 4. Subdivision Surfaces

Subdivision Surface Modeling

Subdivision surfaces combine all the good features of NURBS and polygonal surfaces; they allow organic shapes to be modeled easily and define perfectly smooth shapes with a varying surface.

Realsoft 3D provides a very powerful set of tools for modeling with subdivision surfaces.

This tutorial goes through a number of basic modeling examples for beginners.

Creating a coffee cup

The SDS toolbar allows you to create basic subdivision surface objects.



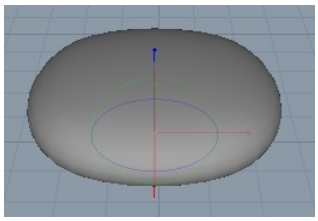
SDS tool bar

Click the SDS/Cube tool button. As usual, the control bar now shows you a number of subdivision cube specific options.



The options for the subdivision surface cube tool

Now switch to the top view either by clicking the top view button in the view control bar at the right side of the view winsow or from the view's popup menu Camera/Top. Define the cube through the view window by entering two points with the mouse. While doing this, you may hold down the Shift key in order to achieve a symmetric cube. A subdivision cube is created.



A subdivision surface defined by a cube-like control polygon

As you can see, the created cube defines an object which doesn't look like a cube at all. In fact, the cube tool created an object, which more or less looks like a sphere!

However, it is relatively easy to turn this spherical object to a coffee cup, or whatever other shape, as we will see soon.

This is actually a very common situation when modeling with subdivision surfaces. One starts from a very simple object, such as a cube, and adds details to it using different subdivision modeling tools.

The new subdivision object should be automatically selected after its creation. If you accidentallyy deselected it, click its name in the select window. As usual, the tool control bar automatically shows you the appropriate subdivision modeling tools as soon as a subdivision object becomes selected.



The control bar shows you the tools, which can be applied to the selected subdivision object

As you can see, most of the tools are disabled. The reason for this is that most of the subdivision modeling tools can only be applied in the 'Edit' mode. Subdivision objects provide three kinds of editing handles: faces, edges and points.



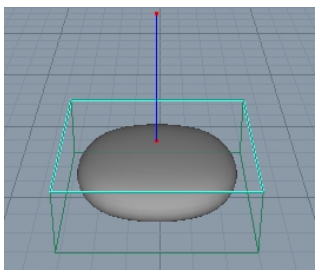
Select Face edit mode

Point handles allow you to single point edit the subdivision object. There are also a number of tools, which can be applied to points. There are also edge and face handles available - they modify several points at a time.

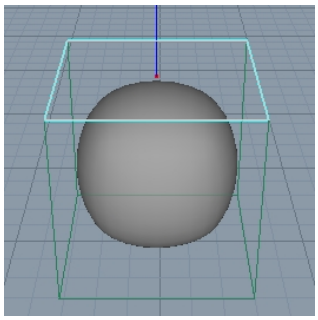
In order to turn our cube into a cup, we will be using the face handle mode. Select Face from the Control Bar's Edit field.

Select the top face by clicking it with the left mouse button. The selected face is highlighted.

Rotate the view window (drag downwards with the right mouse button Alt key held down) so that you can see the object's front and top sides. You can see a so called *face normal handle* sticking out from the selected top face. The face normal handle is perpendicular to the selected face and allows you to move the selected face.



Top face selected



Top face moved along its normal

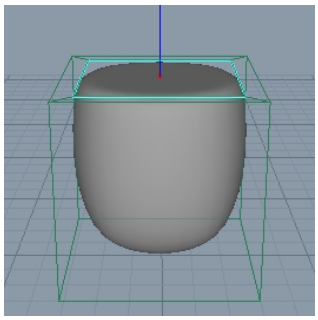
Let's move the face to make our 'sphere' a bit higher. To do this, simply move the mouse over the center of the face normal handle and drag it. Note: don't drag the end points of the handle. They scale and rotate the face.

When you selected the top face, many of the subdivision modeling tools in the control bar were enabled. This indicates that they can be applied to the selected face.

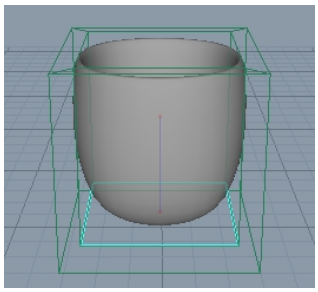
Activate the Subdiv tool (Subdivide Face) and LMB click in the view window.

Subdivide Face tool

When you now move the mouse up/down, you can see the effect of the subdivision tool. When an appropriate subdivision degree is achieved, click the left mouse button again to accept the operation.



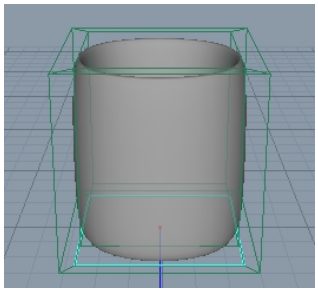
Top face subdivided



Top face extruded inwards

Now activate the Extrude tool and enter one point through the view window. By moving the mouse up/down you can extrude the top face up/down. If you extrude it downwards, the object starts to look like a cup.

The bottom of the cup is too round. Subdividing the bottom face adds new geometry, which will help to flatten the base. So, alt-drag with the right mouse button held down to rotate the view so that you see the cup from below. Click the bottom face to select it. Instead of pressing the Subdiv tool button, let's use the object handles. Hold the Ctrl key down and LMB drag the outer end knob of the face handle line. A new face appears and changes its size while you move the mouse. Make the new face almost as large as the original bottom face and release the mouse.



Bottom face subdivided

As you noticed, many useful subdivision tools can be activated by dragging the handles while holding a suitable modifier key down. The extrude tool which was applied above can be activated by Ctrl-dragging the actual face handle line, not its end knob. The handles usually provide the fastest way to achieve a certain action, because the tool is available at the same place where your focus is.

Now our cup is able to stand stable which is one of the basic requirements for a decent cup.

As you can see, with just a dozen of control points, we can define a cup with a smooth surface.

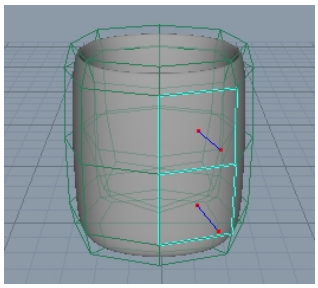
Extruding the handle

Most coffee cups have a handle. Let's see how we can use the Realsoft 3D modeling tools to create one.

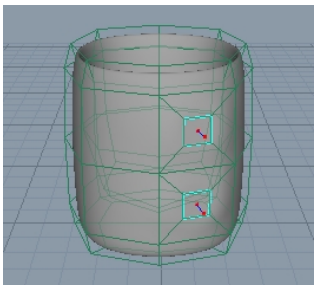
There are numerous ways to create handles for the cup. Which one is the best method, depends on the case. For example, we can use the Extrude tool to extrude two "tentacles" out and connect the ends of the tentacles. Let's try this.

Click the Smooth tool. This subdivides the entire control polygon of the cup, producing a more dense control polygon. However, the actual shape of the cup is not changed by this operation.

Now, select two of the side faces (one on top of the other). Keep the Shift modifier pressed when selecting multiple faces.



Select two side faces and..



..apply the Subdivide Face tool

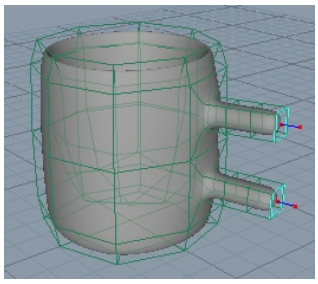
Now apply the Subdiv tool. Subdivide the selected faces just like we did when we subdivided the top face in the beginning of this example.

Activate the Extrude tool. When the tool is activated, the control bar shows you the available extrude tool specific options. Enter 3 into the Sections field in the control bar.

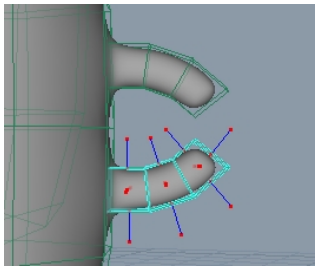


Extrude three cross sections

Now enter two points through the View window to get two tentacles extruded.



Two faces extruded with 3 sections



Extruded tentacles bent

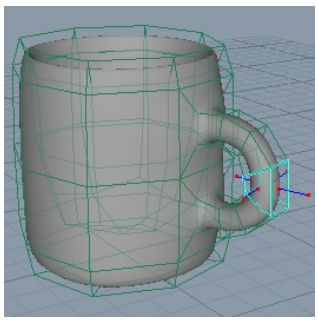
Use the Realsoft 3D Bend tool (in the Transformation toolbar tab) to give the desired shape to the handle tentacles. To do this, rotate the view until you see the tentacles directly from the side. In the view window, LMB-drag a selection box, which includes the faces of the upper tentacle. The faces inside the box become selected. In edit mode, transformations modify only selected handles, and we can now bend only the upper tentacle.

Click the bend tool to activate it. Click once in the middle of the first cross section of the tentacle and click another time at the end of the tentacle. Move the mouse to find a suitable bend and click to finish the tool. Drag-select the bottom tentacle area and bend it, too.

Now we only need to connect the ends of the tentacles to finish the cup. To do this, zoom and rotate the view window to see the end face of the upper tentacle from below. Click to select it. Rotate the view upwards to see the end of the lower tentacle well. Press the Shift key (=multi select modifier) down and click the end face of the lower tentacle. When both faces are selected, the Tunnel tool in the toolbar becomes enabled. Click it to connect the selected faces. The cup is now ready.



The Tunnel tool



Tentacles connected

Using Compass Menus in SDS Modeling

The context sensitive control bar allows you to easily access any subdivision object specific tool. Just select a subdivision object and the control bar shows you the tools, which can be applied to the selected subdivision objects.

However, clicking buttons in the toolbar is not the most productive way to access tools in Realsoft 3D.

The compass menus provide a much faster interface to most modeling tools. Compass menus are context sensitive just like the control bar. In other words, the structure of the compass menu depends on the selected objects. For example, when you select an edge, the compass menu shows you the edge specific tools.

Compass menus are also completely user customizable. You can define any number of compass menus and bind them to any key or key combination. This makes compass menus a very powerful interface to access frequently needed tools.

This chapter demonstrates how the compass menus can be used to increase the productivity in subdivision modeling.

Another coffee cup

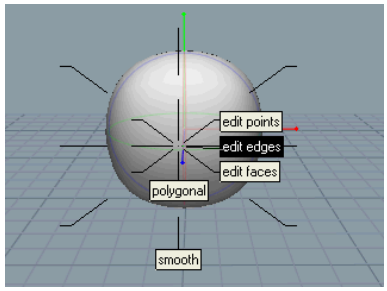
Let's create another coffee cup. This time we will use a slightly different approach.

Tutorial level: Beginner

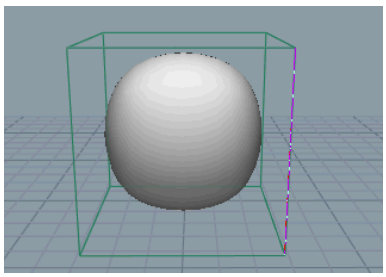
Example project: 'tutorprojects/modeling/sds/cupcreatedin20s'

1. Create a subdivision cube, select it and select the near east compass menu.

This switches the SDS object to edge editing mode.



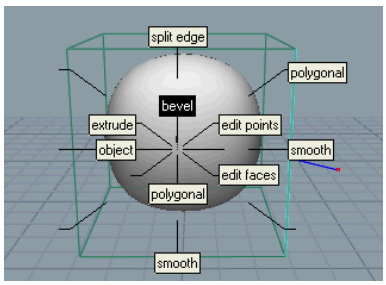
Switch to the edge editing mode by selecting the 'near east' menu



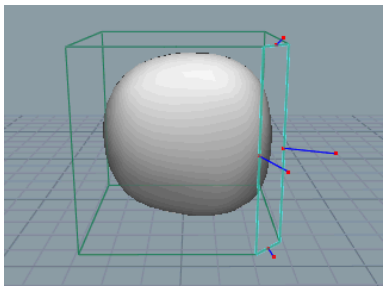
Mouse moved over an edge handle

2. Move the mouse over one of the vertical edges. The edge in question is highlighted and the cursor is changed to indicate that the mouse really hits the edge in question. Click the edge to select it.

Now select the near north menu. This activates the 'Bevel' tool and starts beveling.



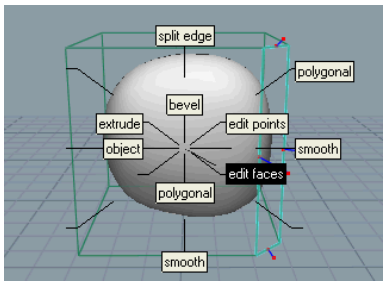
Select an edge and start the Bevel tool



The edge beveled

You only have to move the mouse up/down to control the beveling size. Bevel the edge so that the size of the beveling corresponds to the thickness of the handle to be created. This creates a new face into the cube.

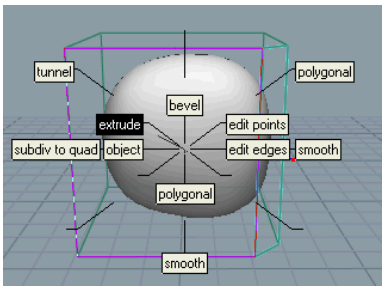
3. Switch to the face mode.



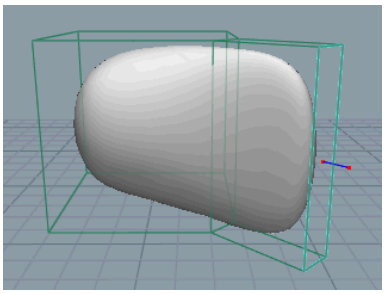
The near southeast compass menu switches to face editing

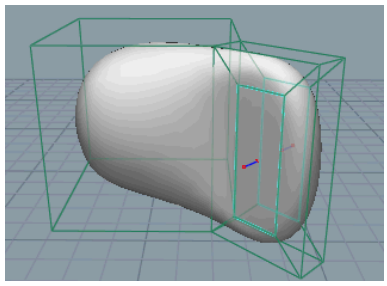
Again, you can use the Edit field in the tool control bar. The same can be accomplished by selecting the near southeast compass menu.

4. Move the mouse over the face so that the face gets highlighted and the cursor is changed. Click to select the face. Then select the near northwest compass menu. Move the mouse up/down to control the level of extrusion.



Select the near northwest compass menu to activate the extrude tool





Extrude a tentacle and subdivide the side faces of the tentacle

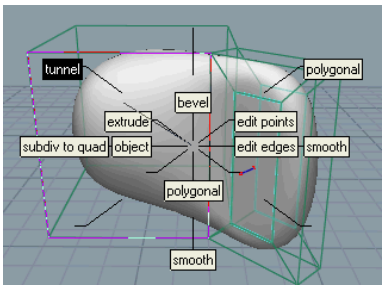
Next we have to drill a hole through the extruded handle tentacle.

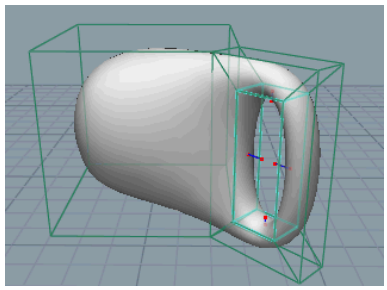
5. Select both side faces of the extruded handle tentacle (select the first side face, rotate the view to the opposite side, hold down the Shift key and click the other side face).

Now, subdivide the selected faces by activating the 'Subdivide' tool from the tool control bar and by entering two points through the view window.

As a result of subdivision, the newly created inner faces become selected. The only thing you have to do is to apply the 'Tunnel' tool to the selected faces.

To do this, select the far northwest compass menu.



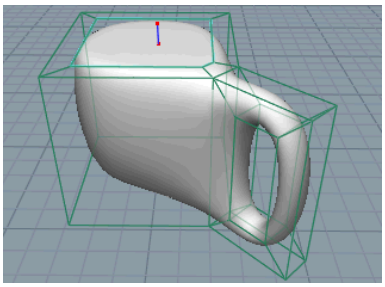


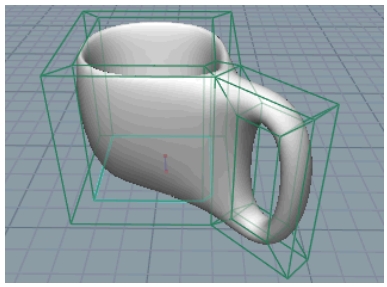
Create a hole through the handle using the Tunnel tool

6. Rotate the view downwards to see the bottom face of the cup. Select it and apply the 'Subdivide' tool to get a flat and stable base for the cup.

7. Rotate the view upwards, select the top face and apply 'Subdivide' to achieve a smaller inner face.

8. Activate 'Extrude' using the compass menu and extrude the inner top face inside the cup to make the cup hollow.





Subdivide the top and bottom faces and extrude the inner top face inside the cup

If you practice this tutorial a couple of times, you will be able to model the cup in less than 20 seconds!

Using Point, Edge and Face Handles

Object handles allow you to apply most of the commonly needed operations simply by dragging the appropriate handle. For example, dragging a point handle allows you to move the selected point.

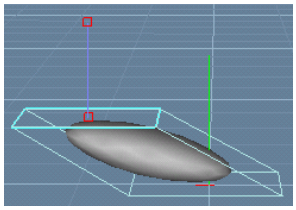
This tutorial demonstrates how the SDS object specific handles can be used to improve productivity in subdivision modeling.

Face handles

Faces can be selected and moved by simply clicking or dragging the face itself.

All the default handle selection rules also apply to face handles. For example, you can multi-select faces by holding down the Shift key while clicking them. You can use the drag box to select multiple faces in one operation.

The comma (",") key is the depth modifier. By holding down the comma key, you can select faces behind rather than in front. In conjunction with the Shift key, one can select faces in the front AND behind - allowing you to easily select back faces without having to rotate the view. For example, if you want to select the 'ear' faces from the side of a head, simply hold down the comma and the Shift keys while clicking the left mouse button. The first click selects the nearest 'ear' face and the second click selects the face on the other side of the head.



An object sheared by moving a face

Face edge handles

The edges of a face can be used for activating the following face tools:

- drag: move on edge (move face in the direction of the edge)

- drag + Shift: scale face about the edge center

- drag + Alt: rotate face about the edge.

Face normal

Face normal handles are only shown for the selected faces. The face normal handle is always perpendicular to the face.

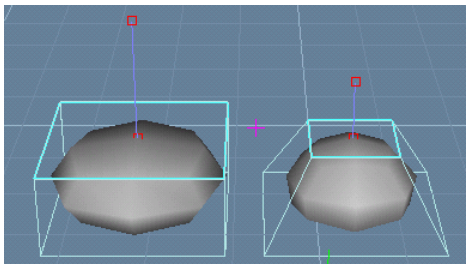
A face normal handle consists of three separate sub handles:

- end point - scales, subdivides
- rotate handle (face center)
- move handle (connecting line)

As you may already know, the face handle allows you to move the face along its normal. We lifted the top face of our coffee cup in the 'create a cup' tutorial with this handle.

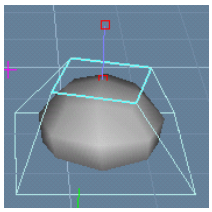
The length of the surface normal handle is relative to the size of the face. If you scale the face smaller, also the length of the surface normal handle gets smaller. Correspondingly, by changing the length of the surface normal handle, the size of the face can be changed.

To do this, simply drag the end point of the face normal handle.



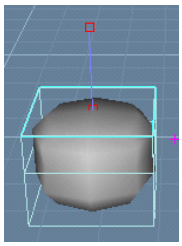
Drag the end point of the face normal handle to scale the face

The other end of the surface normal handle i.e. the *face center handle* allows you to rotate the face.

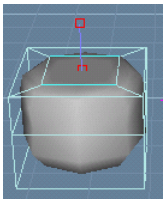


In most operating systems, you can duplicate files, icons, etc. by holding down the Ctrl key while dragging the icon. In Realsoft 3D, you can do the same with geometric objects; holding down the Ctrl key while moving an object duplicates the object.

In addition to entire objects, this works for faces too. Holding down the Ctrl key while dragging a face normal handle duplicates the face and connects the edges of the copied face to the edges of the original face. In other words, it extrudes the face.



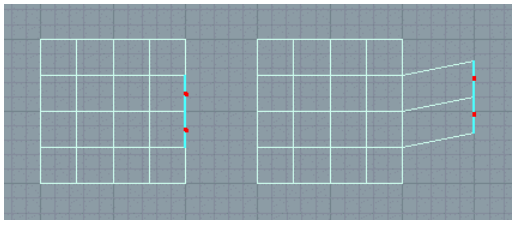
By dragging the end point of a face normal handle (scale handle) with the Ctrl key, you can subdivide the face.



Edge handles

Edges work similar to faces in that they allow easy selection and moving. Simply click or drag the desired edge to select or move it.

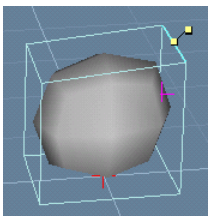
Dragging an edge handle while holding down the Ctlr key copies the selected edges and connects the copied edges to the original edges by creating new faces. The new edges are automatically selected and can be readily moved with the mouse. In short, this can be used to 'grow' new faces to selected edges.



A subdivision object before and after the 'copy selected edges' interactor.

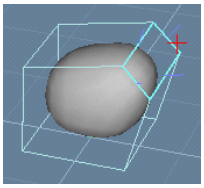
Edge normal

When an edge is selected, its edge normal handle is shown. This handle works similar to a face normal handle: it allows you to apply move on normal, rotate and scale operations to the edge.



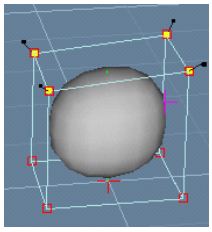
An edge selected and its edge normal handle shown

Edges cannot be subdivided or extruded. Therefore, Ctrl+normal handle is the same as the normal handle alone, and Ctrl+scale handle is used for beveling.



An edge beveled by dragging the end point of the edge normal while holding down the Ctrl key

Point handles



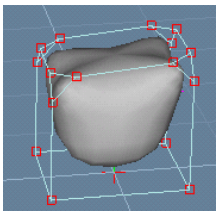
Four point handles selected

Point handles support most of the features described above. Dragging the actual point allows you to move it freely in 3D space.

When a point handle is selected, it shows its point normal handle.

The point normal handle works similar to the face and edge normal handles. It allows you to move the point along its normal.

You can't rotate or scale point handles because a point does not have a size. Therefore, the end point of a point normal handle always applies the bevel vertex tool.



Selected points beveled by dragging the end point of the point normal handle

Pointwise Creation of Arbitrary SDS Shapes

The basic creation tools (rectangle, cube, sphere, circle) allow you to create SDS objects with most faces having four points. However, Realsoft 3D does not set any restrictions on the number of points per face. A face can include any number of points.

This tutorial demonstrates how to use the 'Subdivision' modeling tool to create subdivision objects point by point (or face by face).

The 'Subdivision' creation tool can be found from the toolbar.

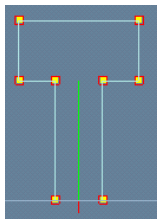


Subdivision creation tool

1. Activate the tool. The control bar now shows you the tool specific options.



Make sure that the Auto Accept checkbox is reset. This allows you to add any number of points to a face. Set Type to Smoothen to NURBS.



A face consisting of eight points

2. Now, enter eight points, which form the letter 'T, then click Accept.

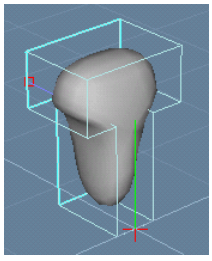
3. Select the created object, switch to the face editing mode and select the face.

4. Activate the Extrude tool. The Control Bar shows you its options.



Set the Leave Floor option

5. Check the Leave Floor option and extrude the face. Our 'T' object is ready.



An object consisting of faces with four and eight points

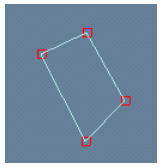
Creating multiple faces

You can also use existing points with the Subdivision creation tool. This can be achieved by holding down the Alt key while entering the points. Then the entered points become snapped to the nearest existing point (new points will not be added to the object).

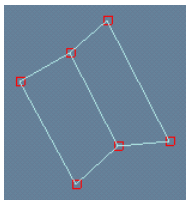
Let's create a head, which consists of multiple faces.

1. Activate the Subdivision creation tool and set the 'Auto Accept' option. Make sure that the 'Points per Face' field is set to '4'.

2. Create the first face by entering four points.



First face defined, four new points created



Second face created, two new points inserted

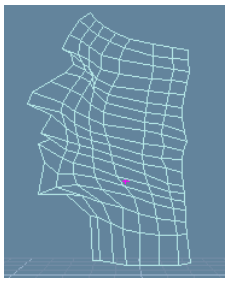
Because we activated the Auto Accept mode, the face is automatically created and you can start to define the next face. Do not click 'Accept'!

3. Now, hold down the Alt key and 'snap' the two first points to the existing points. Then release the Alt key and enter the remaining two points.

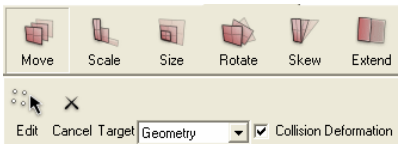
4. Continue adding new faces to the mesh by repeating step 3 so that the mesh finally represents a face from the side. When done, accept the tool. We now have a planar mesh model of one half of a face.

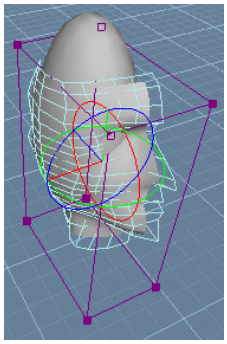
5. Reset the editing mode and create an analytic sphere behind the face. We will use collision detection to deform the face to make it a real 3D surface.

6. Activate the Transformation/Move tool and check the Collision Deformation option. Then move the subdivision mesh towards the sphere to get the face deformed.

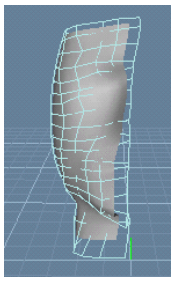


A planar subdivision mesh representing a face





Select the mesh, activate the move tool with collision deformation and move and deform the mesh against the sphere



Left side of the face

7. Switch to the face editing mode again and take the front view.

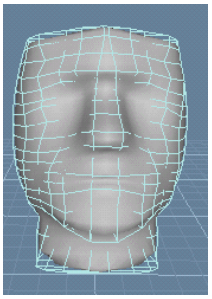
Duplicate tool

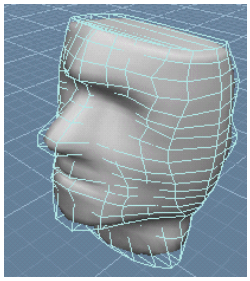
8. Click the Duplicate tool in the control bar. This duplicates all the faces in the object, creates new faces, which connect the edges between the original faces and the duplicated faces, and leaves the newly created faces selected.

9. Activate the Transformation/Mirror tool and mirror the duplicated faces to the right side.

The head is now ready.

Note: this is not the best way to model a head. The purpose was to demonstrate the Subdivision creation tool.





A complete head

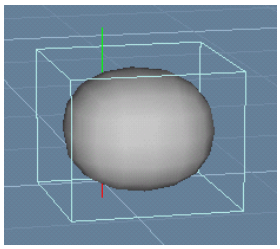
Modeling a Bird

This is a very basic subdivision modeling example which mostly uses two tools: subdivide and extrude.

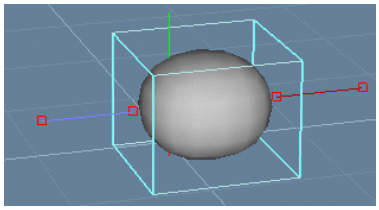
Recall that these tools can be activated as follows:

- *Subdivide*: Hold down the Ctrl key while dragging the face normal end point.
- *Extrude*: Hold down the Ctrl key while dragging the actual face normal line.

You can also activate these tools using the keyboard short cuts, compass or the context sensitive toolbar.



The body of a bird



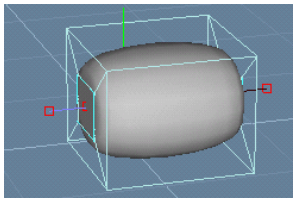
Side faces selected

1. Create a SDS cube and switch to the 'Edit/Face' mode.

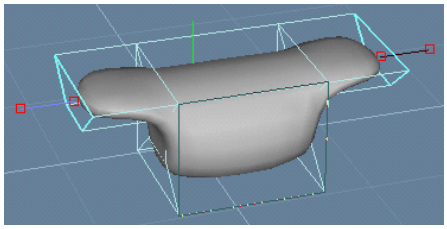
2. Select the left and right side faces. Now you can see their face normal handles.

3. Subdivide the selected faces: hold down the Ctrl key and drag the left normal end handle towards the face center.

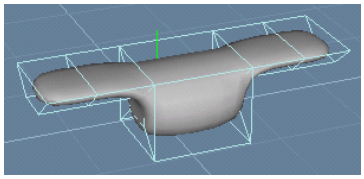
4. Select the topmost left and right faces and drag the 'move on normal' handle while holding down the Ctrl key to extrude. Do this twice.



Faces subdivided



Head and tail extruded

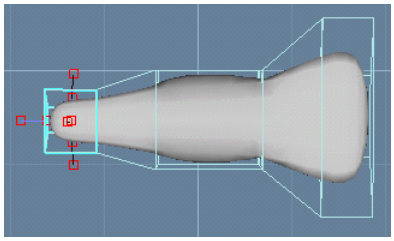


Head and tail extruded again

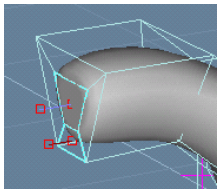
5. Now, switch to the top view and multi-select the right end faces. Stretch them wider (use the Transformation/Extend tool) to form a tail. Then use the Extend tool again to make the opposite end, the head, narrower.

6. Subdivide the end face of the head so that we can extrude a beak out of it.

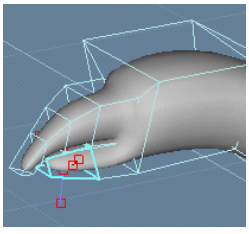
7. Extrude two end faces a couple of times and scale them to look like a beak.



The bird from the top view

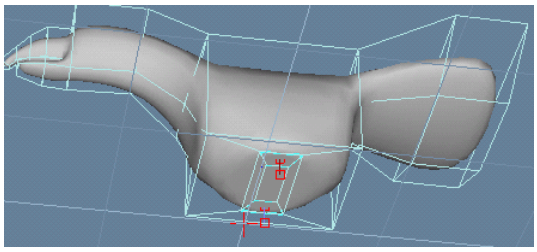


Head end face subdivided and two faces selected for extrusion



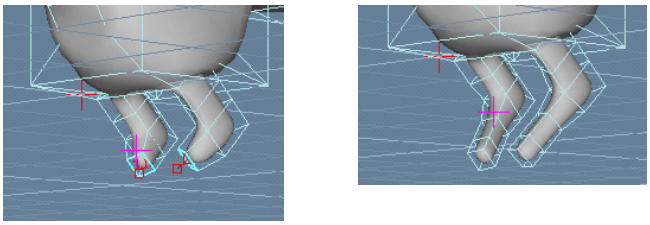
The head finished

8. Subdivide the bottom face of the bird. Subdivide the created middle bottom face again. Scale a suitable pair of the bottom faces smaller so that they can be used for extruding legs out of the body.



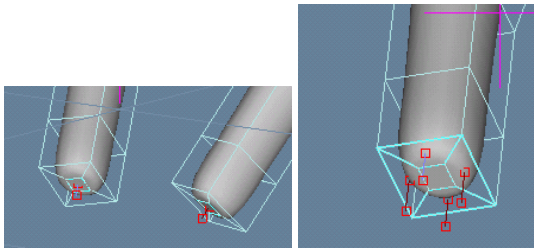
The bottom face subdivided twice and two faces selected for the leg extrusion

9. Now extrude the selected faces a couple of times to get the legs extruded.



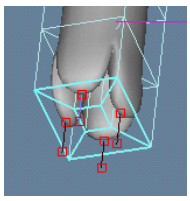
Legs extruded

10. Subdivide the end faces of the legs. Then select the four outermost end faces in both legs for toe extrusion.

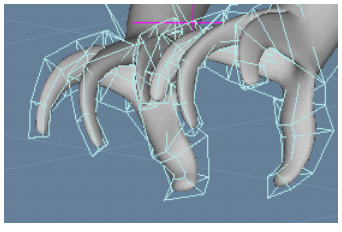


Subdivide the end faces of the legs. Four outermost faces selected.

11. This time start the 'Extrude' tool from the control bar. Set the Region option to Separately and extrude the toes.



Toes extruded



Toes finished

12. Modify the extruded toe faces making three toes to point forward and one toe to point backwards.

The next task is to extrude wings out of the side faces of the body to get our bird completed. However, this would just be repeating the steps described above. Therefore, we complete this example here.

Converting Objects to Subdivision Form

The Convert to SDS tool allows you to convert other geometric objects, such as NURBS surfaces and quadrics, to SDS objects.

Curves are converted to sets of SDS vertices. No faces will be created.

Surfaces are converted to SDS surfaces consisting of triangles (faces with three vertices) and quadrangles (faces with four vertices).

The standard Realsoft 3D installation contains a scanned NURBS head. You can convert it to the SDS form as follows:

1. Go to the select window's geometry tab and select 'Paste from file'. Load the file 'models/head'.

2. Select the loaded head. Activate the To SDS tool from the 'SDS' tab of the toolbar. By default, the tool examines the complexity of the selected object and attempts to generate an appropriate number of faces for the SDS object. This automatic resolution detection is usually desired so just select 'Accept' from the View's pop-up menu to finish the conversion. A SDS head is created.

3. Delete the original NURBS head.



Note

You can also check the 'Del orig.' option in the control bar, which deletes the original object after converting it to SDS. Then step 3 is not needed.

Inverting SDS surfaces

Conversion to SDS form works by evaluating surface points of the selected object to form the control polygon of the SDS object. Therefore, the generated SDS surface is typically slightly smaller than the original surface. A SDS surface usually shrinks inwards from its control polygon. The shrinking can be compensated by appying the 'Invert' tool. The actual surface will then pass through the previously used control points.

Converting polygonal models to the SDS form

The following three tools can be used to achieve a successful conversion from polygonal models to SDS surfaces:

Convert to SDS

The To SDS tool handles polygonal (typically triangle based) models by simply converting the polygons to corresponding SDS faces. Polygonal models often consist of triangles. Unfortunately a non-symmetric triangular structure, when used as a SDS control polygon, usually gives a very poor rendering quality. The triangular nature of the model can be seen from the rendering.

Merge Triangles to Quadrangles

The 'Merge Triangles to Quadrangles' tool scans through the selected triangles (or the whole object if it is not in the editing mode) and attempts to merge them to quadrangles. Whenever two associated triangles form an approximately rectangular face, the tool merges the triangles to one rectangle. This simplifies the model. Also, quadrangles usually produce much nicer SDS surfaces.

Note that often it is not possible to merge all the triangles.

The tool can be found from the main menu bar: 'Methods/Sub Division Surface' submenu includes a full SDS specific toolkit.

Invert

Finally, you can invert the generated surface so that the surface passes through the original polygon points.

Create a coffee cup

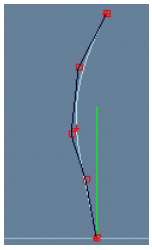
Any geometric object that defines a true surface can be converted to a subdivision surface. You can use, for example, all NURBS modeling tools to create subdivision objects easily. This is demonstrated in the following tutorial.

Let's first create two simple NURBS surfaces, which represent the body and the handle of a coffee cup.

1. Create a vertical curve representing the rotation axis of a coffee cup. To do this, select the NURBS/Curve tool and click the left mouse button in the view window. Finish the curve by selecting Accept from the view's pop-up menu.



Nurbs Curve tool



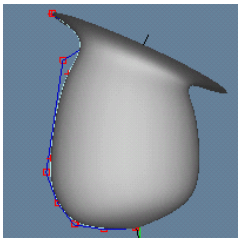
Points of the curve

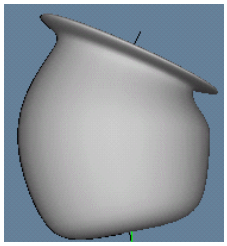
2. Select the created axis curve and activate the NURBS/Curve tool again. When the curve tool is active, the control bar shows you a number of curve specific creation options. Set the Pen option to Rotate.



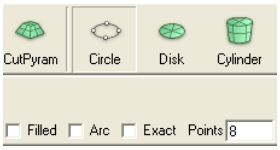
Activate the Curve tool and set Pen to Rotate

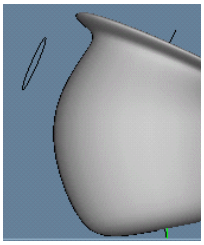
3. Enter a number of points through the view window to define the profile curve for the cup. You can see the rotated surface in real time. Accept when the profile is ready.





Define a profile curve for the cup

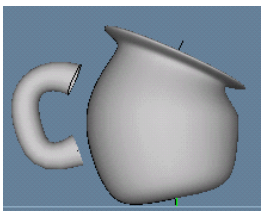




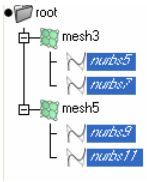
4. Switch to the side view and create a circle representing the cross-section of the handle. For example, select Nurbs Circle tool from the NURBS tools menu. Then enter two points through the view window to define the center and the radius for the circle.

Create a NURBS circle representing the cross-section of a handle

5. Select the created circle, activate the NURBS/Curve tool again and set the Pen option to Sweep. Enter a number of points through the view window to create a handle for the cup. Then click Accept.



Two NURBS surfaces

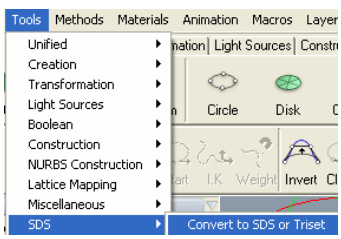


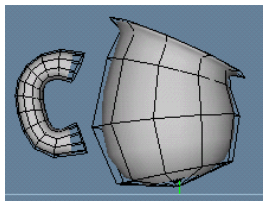
Delete the construction curves

Now the NURBS modeling specific part of this tutorial is done and we can convert the NURBS objects to subdivision objects.

Before we do this, delete the construction curves, which were used to create the surfaces. These curves are located inside the NURBS meshes and allow you to edit the shape of the meshes. However, we only want to convert the surfaces to subdivision objects. So, select the curves and delete them.

6. Select both meshes and select the pull-down menu Tools/SDS/Convert to SDS or Triset. This activates the conversion tool. The control bar allows you to define the number of faces to be created. Default options are OK, so just click the Accept button. Two subdivision objects are created. Delete the original NURBS surfaces.





NURBS meshes converted to SDS objects



Note

You can also check the Del orig option in the control bar, which deletes the original objects after converting them to SDS.

We now have to weld the handle to the body of the cup. To do this, we need to merge the two separate SDS objects into a single object.



Merge O tool

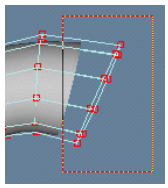
7. Select both SDS objects and click the Merge O (merge objects) button in the control bar. This merges the two selected objects into a single object.

We could now blend the handle to the cup by creating faces manually between the edges of the handle and the body of the cup. However, this is quite a time consuming job. We use the 'Tunnel' tool instead. This tool allows you create a surface through any two faces. The only problem is that there is no face in the ends of the handle surface. This problem is easy to fix:

8. Switch to the point editing mode and select all the points of the upper end of the handle. Then click the Add button to create a closing cap face. Repeat the operation also for the lower end of the handle.

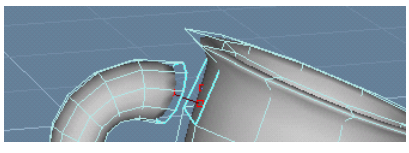


Add face tool



Select the end points of the handle and click the Add tool

9. At this point, make sure that the handle is placed so that the end faces of the handle match well two faces of the body. If not, select all points of the handle, move them from the front view and rotate them from the top view. To deselect points, press the Alt key and click on, or drag around, the points you wish to deselect. Pressing Ctrl + t deselects all selected points.



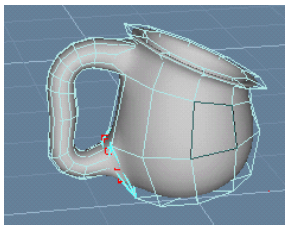
Select the faces to be welded...

10. Now switch to the face mode and select the upper end face of the handle. Hold down the Shift key and select a suitable face from the body of the cup. Then click the 'Tunnel' tool to create a surface through the selected faces.



...and click the Tunnel tool

Repeat this for the second end of the handle.



The handle merged to the cup

Defining Pointwise Properties

You can define surface properties of SDS objects as you would define those of any geometric objects: using materials and constant attributes listed in the 'Col' tab of the property window. In addition to this, you can also set surface properties for individual control points of SDS objects.

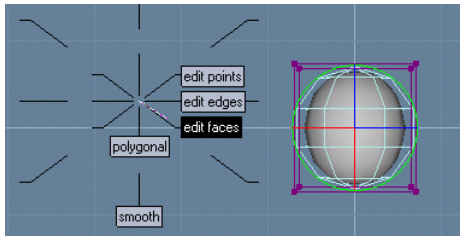
You can define surface properties through selected points, edges and faces. This tutorial demonstrates how to manage pointwise properties.

Pointwise colors

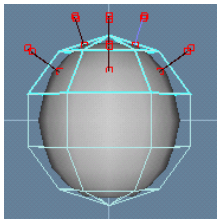
Let's create a sphere whose north pole is red and south pole is blue.

1. Take a Top view and then create a SDS sphere.

2. Take a front view and switch to point, edge or face editing mode (for example, select the 'Edit Faces' from the compass menu).



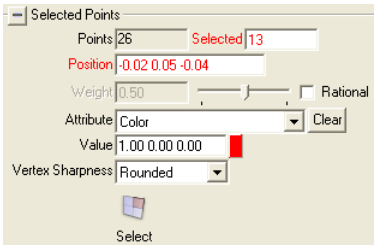
Enter Edit Faces mode



Northern faces selected

3. Select the faces of the upper half of the sphere.

4. Open the property window and go to the 'Spec' tab. Make sure the 'Attribute' field in the 'Selected Points' group is set to 'Color'. Then enter the value '1 0 0' (red) to the 'Value' field.

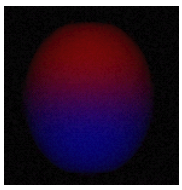


Color property for the northern faces set to red

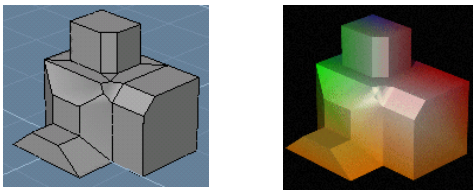
5. Select the southern faces of the sphere and define a blue color through the 'Value' field.

6. Render the subdivision sphere to see the result.

You can define all surface properties, such as transparency and refraction factor, this way.



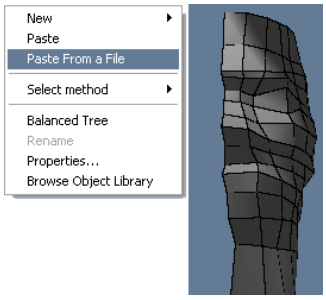
A smooth SDS sphere with pointwise colors



A polygonal SDS object with pointwise colors

Creating Symmetric Objects with the Duplicate Tool

Many real world objects are symmetric in some way. For example, left and right sides of a human head are quite similar. The SDS 'Duplicate' tool is provided for utilizing this observation in SDS modeling. You only have to model 50% of the object and use a transformation tool to get the other part modeled.



Load 'tutorprojects/sds/halfhead'

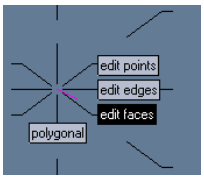
The Duplicate tool takes care of three things. It first duplicates all points, edges and faces in the selected object. Then it finds all open edges in the original and duplicated faces and connects them with an appropriate number of new faces. Finally, the tool selects the duplicated faces so that you can immediately move, mirror, scale or otherwise transform them.

1. Go to the Select Window's geometry tab and click the right mouse button. A pop-up menu is opened. Select Paste from a file option and load the 'tutorprojects/modeling/sds/halfhead' object.

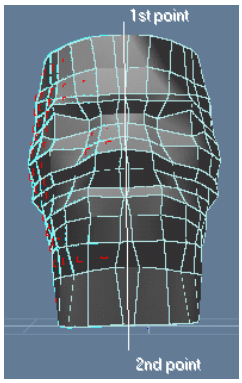
2. Select the loaded face and enter 'Face' mode by selecting the near southeast compass menu.

3. Click the Duplicate tool.

 *Duplicate tool*



Enter face editing mode



Duplicated faces mirrored

Note: Duplicate tool automatically selects the duplicated part of the object. Be careful not to change the selection, because re-selecting the exactly overlapping faces is very difficult. If you accidentally change the selection, undo the 'Duplicate' operation and execute it again.

4. Activate the Mirror tool; it's located in the Transformation tab.

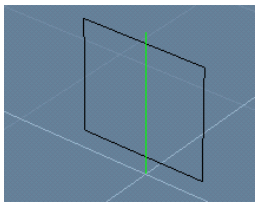
 *Mirror tool*

Then enter two points through the view window to define vertical mirroring axis.

The result is a complete face. The tool connects also the back, top and bottom sides of the face. If this is undesirable, select the connecting faces using the top view and apply the 'Destroy F' tool to them.

A cube

A cube is an object, which clearly has an axis of symmetry: its left and right side faces are identical. So, we only have to create one face and get the other face by using the Duplicate tool. Side faces are created automatically by the tool.



Enter two points through the view window to create a SDS rectangle



Activate the SDS Rectangle tool...

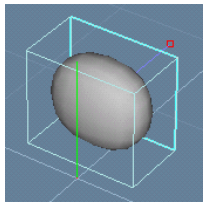
1. Create a SDS rectangle consisting of 2x2 points.

2. Select the rectangle and enter the face editing mode. Now the control bar shows you all the SDS  specific tools. Click the Duplicate tool.

This duplicates the single face of the rectangle object and creates four new faces to connect the edges of the original and the new face.

Note: the tool automatically selects the duplicated face.

3. Activate the Transformation/Move tool and enter two points through the view window to displace the duplicated face from its original position.

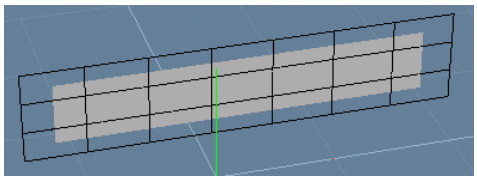


The duplicated face moved

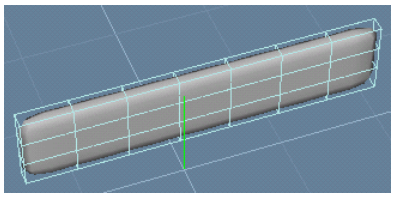
Extruded mesh

In this example, we create an object used by body builders to train their biceps.

1. Activate the SDS/Rectangle tool. This time set the resolution to 8x4 through the control bar (U=8, V=4).

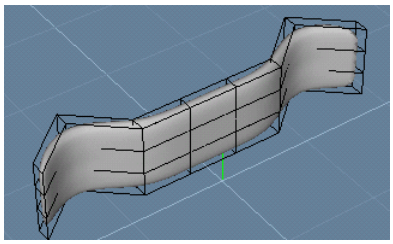


SDS mesh consisting of 8x4 control points



Extruded mesh

2. Enter Edit/Face mode and click the Duplicate tool. Use the Move tool (or one of the normal handles) to displace the duplicated faces from the original faces.



Bent mesh

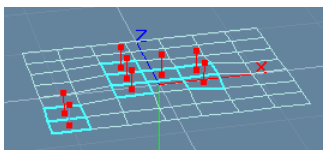
3. Switch to the point editing mode. Select both cross-sections next to the ends of the object and move them backward to bend the object as shown below.

A mesh with holes

The Duplicate tool finds all open edges in the surface to be extruded, including 'holes':

1. Create a SDS rectangle consisting of 8 x 8 control points.

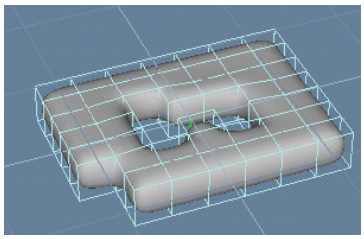
2. Enter the face editing mode and delete a couple of faces from the mesh using the 'Destroy F' tool.



Select a couple of faces and destroy them with the Destroy F tool

Destroy Face tool

3. Click the Duplicate tool and use the Move tool to move the duplicated faces apart from the original faces.

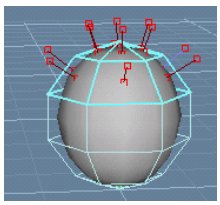


A mesh with holes extruded with the Duplicate tool

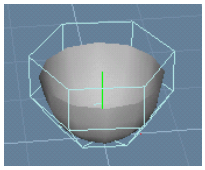
A coffee cup

In this example, we create a coffee cup by modeling the outer surface for the cup first. The inner surface for the cup is created with the 'Duplicate' tool (the inner surface is otherwise identical to the outer surface but slightly smaller).

1. Take a top view and create a SDS sphere. take a front view and select the faces near the north pole as shown in the image.



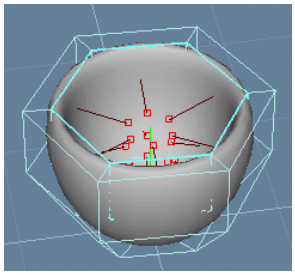
Select the topmost faces



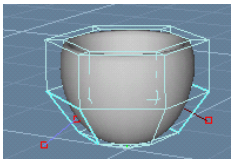
Upper faces destroyed

2. Apply the 'Destroy F' tool to the selected faces.

3. Click the 'Duplicate' tool. This creates the inner surface for the cup. Drag one of the scale handles at the end of normal handles to scale the inner faces smaller.



Outer faces duplicated and scaled inwards to form the inner surface for the cup

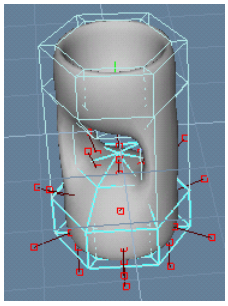


Select two opposite faces

We could create a handle for the cup as demonstrated in some of the earlier tutorials. However, let's be more creative this time. The cup has a round base, and therefore it is not a practical cup to use. We can duplicate and mirror the cup in vertical direction to create a proper base for it.

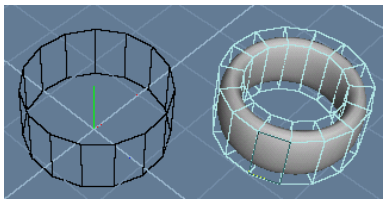
4. Select two opposite faces near the bottom of the cup.

5. Click the 'Destroy F' tool to delete the selected faces. This creates two rectangular holes in the outer surface of the cup. Click 'Duplicate' and then use the 'Mirror' transformation tool to flip the duplicated faces under the cup as shown in the image on the right.

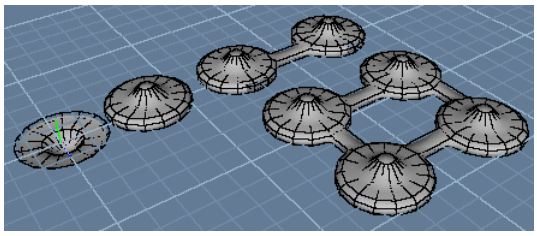


A cup base created by mirroring the duplicated faces

Other examples



Duplicate + Scale

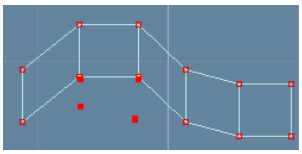


Duplicate + Mirror

Adding New Points and Faces

You can add new points to a SDS object by using the standard Ctrl + drag method. In other words, hold down the Ctrl key while dragging a vertex with the left mouse button.

This duplicates and moves all the selected points.



Two points duplicated by dragging them while holding down the Ctrl key

Creating faces through existing points

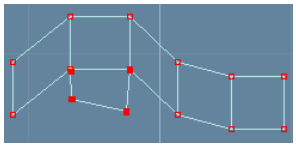
To create new faces through existing points:

1. Switch to the Point editing mode.

2. Select three or more points in the order in which you want them to be inserted to the new face.

Add face tool

3. Click the Add tool.

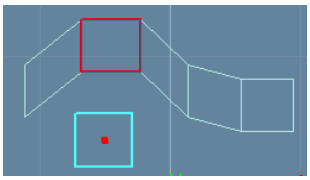


A new face added to the selected vertices

Copying faces

You can copy faces similar to points - hold down the Ctrl key while dragging a face.

This duplicates and moves the selected vertices or faces.

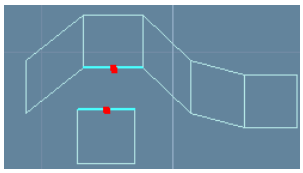


A new face created by dragging an existing face with the Ctrl key

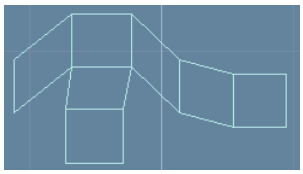
Adding new faces through edges

You can create new faces also through existing edges by using the Add tool. The tool creates a new face through the selected edges. For example, to connect two edges belonging to separate faces:

1. Switch to the Edge mode.
2. Select the desired edges.
3. Click the Add tool.



Two edges selected



Add Face applied

Creating new faces on the fly

The Add face tool allows you to add new faces through existing edges or points. However, sometimes there are no suitable points or edges from which the new faces could be created. You can solve this problem by first creating the points (by dragging existing points with the Ctrl key as described earlier) and then using the Add tool to create a face through the points.

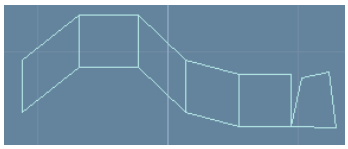
The Faces tool speeds up this process by allowing you to enter new points and faces simultaneously. This tool also allows you to snap to existing points.



Faces tool

For example, if you have a 5x2 SDS mesh and you would like to make it 6x2 mesh by inserting a new face to the tail of the mesh:

1. Select the mesh to be modified.
2. Click the Faces tool of the control bar.
3. Click the left mouse button twice to create two new points for the SDS mesh. Then press down the Alt key and move the mouse near the existing edge points to attach the remaining two points. By holding down the Alt key, you can snap to existing points and no new points will be inserted to the selected SDS object.



A new face under construction

4. When all the four points are inserted, select Accept to terminate the face creation.



Note

If you want to create several faces which all will have equal point count, you can use the Auto Accept mode. For example, to create faces with four points, set Points per face to 4. A new face is created every 4th entered point.



A new face is created automatically after every 4th entered point



Note

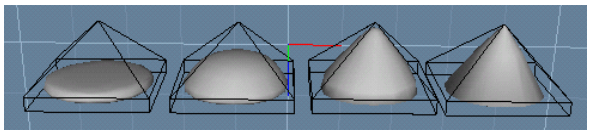
If you only want to insert new points to the selected object (not faces), reset the Create Face option.

Rational Subdivision Surfaces

Rational subdivision surfaces introduce a new parameter called 'weight'. The higher the weight, the stronger the control point affects the surface.

You can turn any SDS object into its rational version by dragging points, edges or faces with the left mouse button while holding down the Alt key. By moving the mouse up, you can increase the weight. By moving the mouse down, you can decrease the weight.

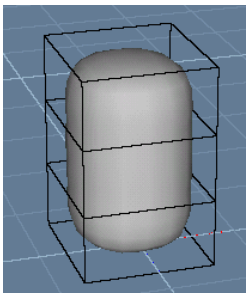
The following image demonstrates the effect of changing the weight of the apex of a pyramid.



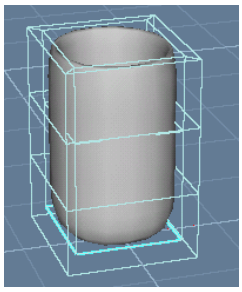
Weights from left to right: 0.1, 1, 10, 100

In the next tutorial we create a simple 'jug' object which demonstrates the power of rational SDS surfaces.

1. Switch to the top view and create a SDS cube of the resolution 2x2x4 (click the SDS/Cube tool and enter U=2, V=2,W=4 through the control bar). Switch to the face mode and subdivide and extrude the top face as shown below. We have created so many cups by now that we don't need to repeat the detailed steps here anymore.



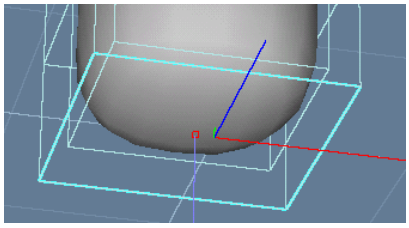
A SDS cube with 2x2x4 control points

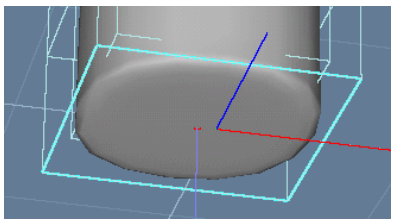


The top face subdivided and extruded inside the cube

The first problem is that our jug has a pretty round base and it wouldn't be able to stand on a table. We could subdivide or bevel the bottom face to get it more 'flat'. However, increasing the weight of the bottom face has the same effect.

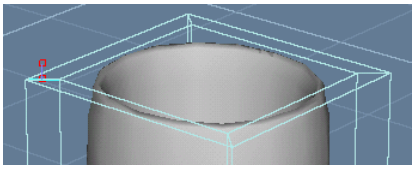
2. Select the bottom face.





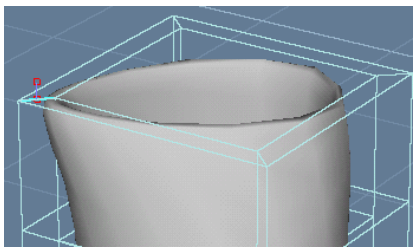
Bottom face weight increased

3. Press down the Alt key and drag the bottom face with the left mouse button. Drag upwards a couple of times until the desired sharpness is achieved.



Select one of the top face edges

Let's create a spout for the jug next.



Weight of an edge increased

4. Switch to the edge mode and select one of the diagonal 'top face' edges.

5. Zoom in so that you can clearly see the edge and drag the edge with the left mouse button while holding down the Alt key. Instead of moving the edge, the weight of the edge is modified. Drag up to increase the weight until the desired sharpness for the spout is achieved.

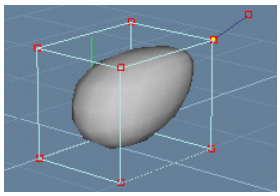
Defining weights through the property window

You can also make the surface rational by setting the Rational checkbox in the Property Window/Spec tab.

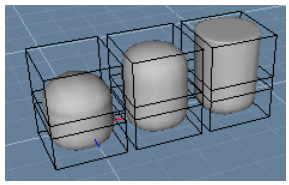


Set the Rational check box in the property window

You can then select any of the control points, edges or faces and define the desired weight through the Weight field.



Weight of the selected point increased



Top face with different weights

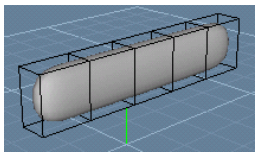
Create a propeller

In this tutorial we demonstrate weights by creating a very simple propeller.

1. Create a SDS cube consisting of 6x2x2 control points.



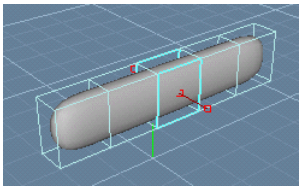
Options for creating a 6x2x2 SDS cube



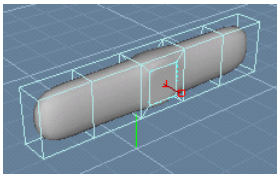
6x2x2 SDS cube

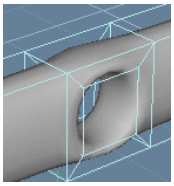
2. Enter the face editing mode and select two faces as shown in the figure:

3. Subdivide the selected faces (for example, hold down the Ctrl key while dragging an end point handle of a face normal). Then click the Tunnel tool to drill a hole for the screw shaft.



Two opposite faces selected for the 'axis' hole

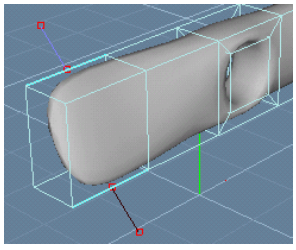




Subdivide and Tunnel tools applied

4. Switch to the edge mode and select two opposite edges at both ends of the screw. Then open the property window, set 'Rational' check box and enter the value 5 to the 'Weight' field.

5. Repeat the step 4 at the other end of the propeller, but this time use the other edge pair to twist the shape to the opposite direction.



Weights of two opposite edges increased

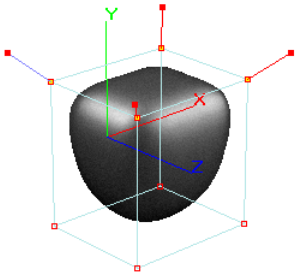
Creases

Creases allow you to create sharp edges for SDS surfaces.

Creases are similar to 'weights' in that they allow you to control how strongly a certain point, edge or face 'attracts' the actual surface.

Realsoft 3D supports three crease levels: sharp, semi sharp and rounded. In addition to this, there are two kinds of creases: point creases and edge creases.

Point creases are affected whenever you use the 'Sharpness' tool in the point editing mode. If you are in edge or face editing mode, edge creases are affected. This means that the result of setting creases for all points of a face is different from setting crease values for the edges of the face.

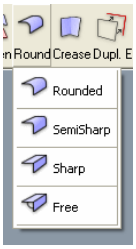


The four top points have a semi sharp crease value. The bottom vertices have the default smooth crease value.

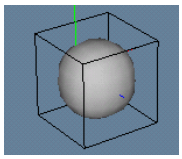
The following tutorials demonstrate how different crease values and types affect the shape of the actual surface.

Point creases

You can define crease values for selected points using the 'Sharpness' SDS tool.



Sharpness tool in the control bar

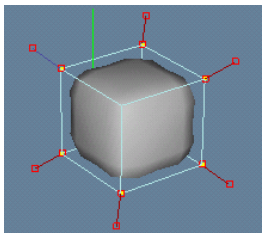


A SDS cube with default crease values

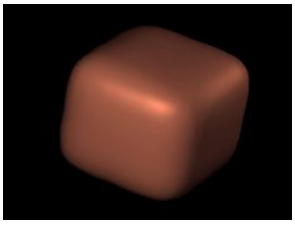
1. Create a SDS Cube. By default all crease values are set to 'rounded' and the resulting surface looks like a sphere.

2. Enter the point editing mode and select all points of the SDS object. The 'Sharpness' control in the tool control bar becomes enabled.

3. Set sharpness to Semi Sharp.



The SDS cube with semi sharp point creases



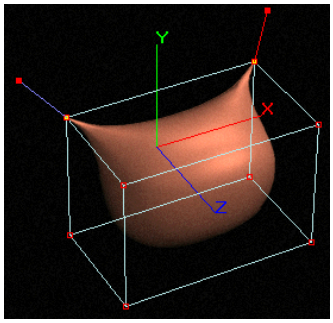
A ray traced SDS cube with semi sharp point creases



Note

OpenGL shading can represent the true shape of a SDS surface only with a limited quality. In photo realistic rendering, creases become rendered with a proper quality.

If you set a point crease to sharp, you'll get a sharp peak, as shown in the image on the right.



Two points with a sharp crease

Edge creases

Edge creases are defined in edge or face editing modes. Defining a crease value for a face corresponds to defining a crease value for all edges of the face.

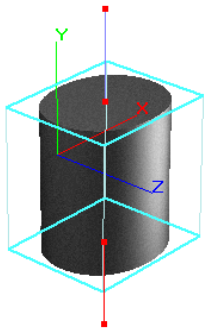
Let's see how edge creases differ from point creases.

1. Create a SDS cube just like in the previous example.

2. This time, enter the face mode and select the top and bottom faces.

3. Set crease to 'Sharp' using the sharpness tool.

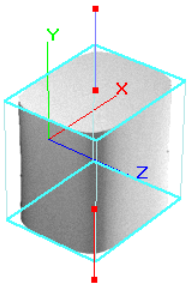
The actual surface looks like a cylinder now.



A SDS cube with 'Sharp' top and bottom face creases

Mixing point and edge creases

Point and edge creases can be mixed.



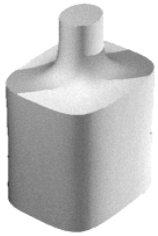
The edge creases of the top and bottom faces are 'sharp'. All point creases are set to 'semi sharp'. Side edges are also semi sharp.



Note

If you apply tools which create new points or edges for SDS object, all inserted points will have a default 'smooth' crease value.

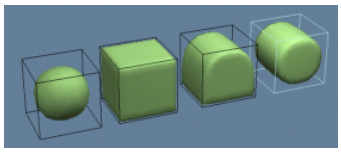
For example, if you subdivide and extrude the top face of the SDS object in the image above, the following object is the result:



Top face subdivided and extruded

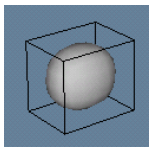
Rounding

The rounding feature allows you to round edges with a desired rounding radius. Rounding is different from creases in that it can only be applied to edges. The biggest advantage is that the size of the rounding is freely adjustable, whereas creases have only three different values (sharp - semi-sharp - rounded).



Four identical SDS cubes with different edge rounding

Rounding is an edge specific attribute, which can be set in the edge editing and face editing modes. Rounding a face corresponds to rounding all the edges of the face.



SDS cube with no edges rounded

1. Create a SDS cube. By default, no rounding values are defined for the edges and the surface looks like a sphere.



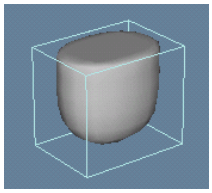
2. Switch to the face editing mode and select the top face. Open the property window and set the 'Edge Sharpness' to 'Free'. This allows us to specify exactly the desired rounding radius. Change the 'Edge Rounding' field in the 'Spec' tab to 0.001 meters. In other words, define 1 mm rounding for all the edges of the top face.



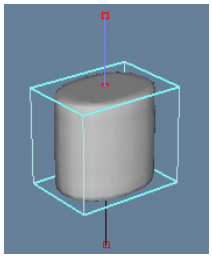
Note

You may have to set the number of decimals to a value of 3 in the Preferences Window - Metrics tab to be able to enter the value of 0.001.

The cube now defines the following surface:



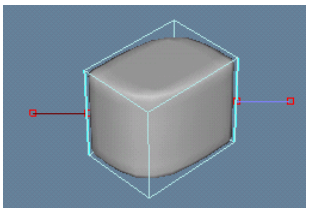
Top face with a 0.001 m rounding



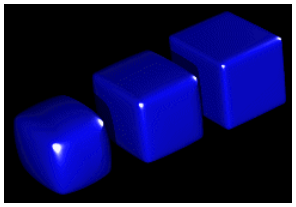
Edges of the top and bottom faces rounded

3. Now, select the bottom face and set the rounding of it to 0.001 as well. You now have a cylinder whose both ends are slightly rounded.

4. Switch to the edge editing mode, select two opposite vertical edges and define a 0.001 rounding for them as well.



0.001 rounding defined for two opposite vertical edges



Identical SDS control polygon cubes with different rounding values

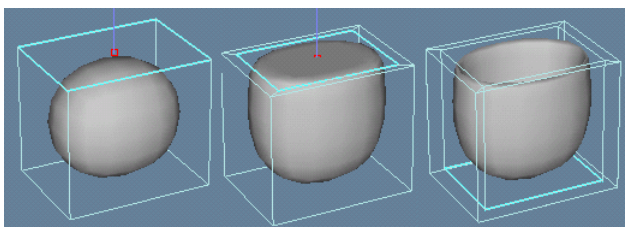
It is already easy to guess what happens if you define rounding for all edges: the result is a cube with rounded edges.

A box with rounded edges

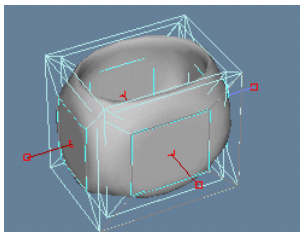
Let's create a box with rounded edges.

1. Create a SDS cube.

2. Switch to the face editing mode and select the top face. Subdivide and extrude the top face inwards as shown in the figure below.



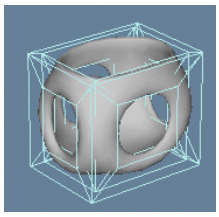
Select, subdivide and extrude the top face



Vertical faces subdivided

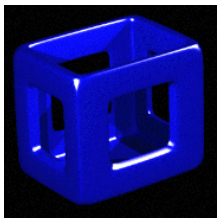
3. Now, select all the vertical faces (including the inner side faces) and subdivide them. Make sure you set the 'Region' to 'Seperately' in the Subdivide Face tool control bar.

4. Select a face pair in the middle of a cube 'wall' and apply the tunnel tool. This drills a hole through the wall. Repeat this for all four sides of the box.



Walls punched with the tunnel tool

5. Use the drag box to select all faces and set a 0.001 meter rounding radius through the property window. The box is now ready.



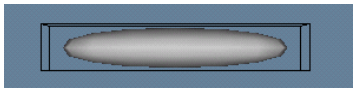
Edges with roundings

Model a sink

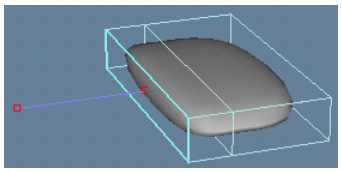
Let's create a beautiful sink. Similar to those you can find in any modern bathroom.

Example project: 'tutorproject/modeling/sds/sink'

1. Switch to the front view and create a SDS cube representing the sink from the front.



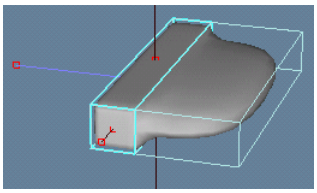
A SDS cube in the front view



One of the faces extruded

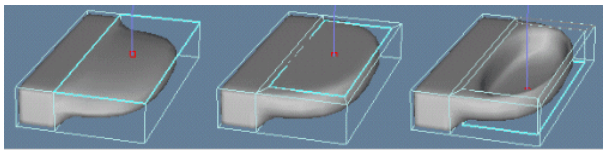
2. Enter the face editing mode and extrude the 'back face' of the SDS cube.

3. Use dragging to select all the faces in the extruded part and define a small rounding (e.g. 0.002 meters) through the property window.



0.002 rounding defined for the selected faces

4. Select, subdivide and extrude downwards the remaining 'top' face of the sink as demonstrated by the attached image.



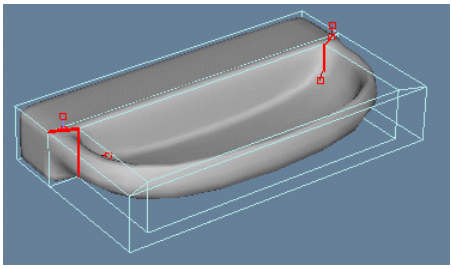
Rightmost top face selected, subdivided and extruded downwards

5. Switch to the edge editing mode and round the four edges shown in the image to value 0.002.

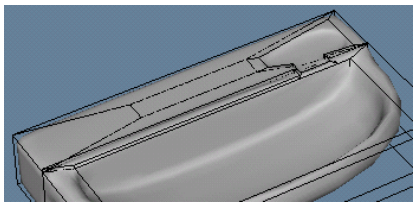


Note

It can be difficult to select edges in the shaded mode because the edges are partially hidden by the actual surface. You can switch to wire frame rendering mode to select the edges easily.



Edges marked with red color rounded



Top face subdivided. One of the new faces subdivided and extruded inwards to form a place for soap.

Our sink starts to look like a real sink already. However, all real sinks have some additional details like a place for soap. Use your imagination to further edit the sink. For example, to create a place for a soap by subdividing and extruding.

After some practicing, you will be able to create a real sink in just a couple of minutes.



The finished sink

Assigning Materials to Faces

You can attach textures and materials to SDS objects in several ways:

- Using geometric mapping objects, such as the parallel and the sphere mapping. This method is suitable for procedurally defined (3D volume) materials. The disadvantage is that rigid mapping coordinates cannot follow deforming SDS objects in animations.
- With a default mapping object, which uses a vertex channel to define the texture or shader coordinates. The number of vertex channels is not limited: in addition to the UV Coordinates channel, which is available in every SDS object, the user can add his/her own channels 'myuv1', 'myuv2' etc. The disadvantage of vertex channels in surface texturing is that it is often difficult or impossible to unfold a complex, branching surface into a UV plane continuously enough. Two points that are close in 3D space may get dislocated in UV space.
- With face materials or textures. These are attached using a private set of UV coordinates for each face, which approach solves the unfolding problems of vertex channels. On the other hand, interpolation of face material UVs is less continuous as vertex channel interpolation, because neighbor faces do not contribute.
- With UV set objects. UV sets are sub meshes of SDS objects defined in UV coordinate space. You can use any number of UV sets to split the parent mesh geometry into suitable UV areas (e.g into front and back projections). Each UV set can contain any number of disconnected face groups.

Assigning Materials to Faces Using Drag & Drop

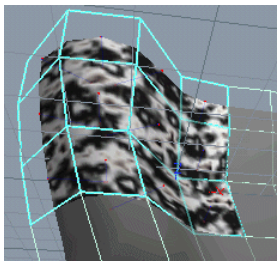
The easiest way to attach materials to faces is using drag & drop.

1. Put the SDS object into the face edit mode.

2. Select the target faces

3. Zoom and rotate the view so that target faces are relatively parallel to the view window and fill it. UV coordinates for face materials will be the projection of vertices into the view.

4. Drag & drop a suitable material into the view window. This assigns the material to the selected faces.



Granite material assigned to the selected faces

You can assign any number of materials to a single face. Materials are evaluated in the order in which they are assigned.

Editing Texture Coordinates in UV Editor

Each material gets an unique set of texture coordinates through which you can control the position and size of the texture. You can edit these texture coordinates in the UV Editor window.

The UV editor allows you to edit the UV coordinates of the currently selected face material (there can be any number of materials per face, but the *current face material* allows you to select, which one of them is shown in the UV editor).

When you open the UV editor for the first time, it automatically evaluates the current face material to a bit map image which is then shown as a backdrop image in the UV Editor.



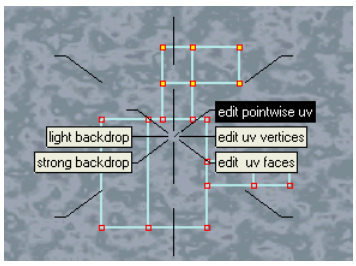
Note

Evaluating the backdrop image can take several seconds depending on the size of the UV editor window. However, this process is run as a separate thread so you can start using the UV editor as soon as it opens.

Through the UV editor you can edit the UV coordinates of the currently selected face material. Use any transformation or deformation tools, such as Scale, Rotate, Mirror, Bend etc. to modify the UV coordinates.

UV editor also supports compass menus. You can put frequently needed UV manipulation tools (mirror faces etc.) to compass menus. The default startup project defines an example compass menu for the SDS objects.

Pressing the Delete key on the UV Editor window removes the current face material from the selected faces (deletes the selected UV faces).

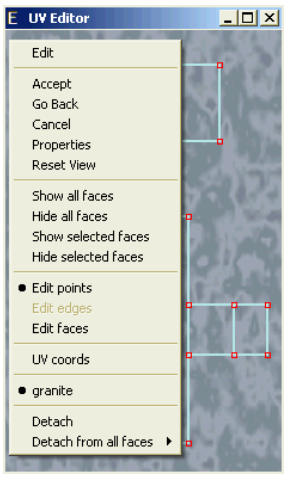


The UV editor and a SDS specific compass menu

Most of the standard modifiers also work in the UV Editor. For example, all selection modifiers (Shift = Add to selection, Ctrl = Intersection, Alt = Remove from selection) etc. work. When modifying faces, Shift activates angle grid and so on.

When modifying the UV coordinates in the UV Editor, 3D view windows update in real time.

There is also a context sensitive pop-up menu associated with the UV View.



Just like in 3D views, there are three editing modes available: Points, Edges and Faces. However, edge editing is not avalaible for face materials, only for vertex channels.

The following menu options control which faces are shown on the UV Editor (note that only faces that contain the selected face material can be shown):

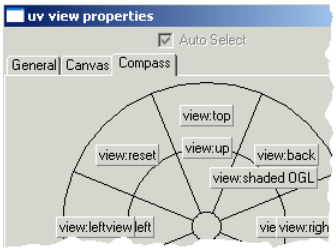
- "Show all faces" - Shows all the faces that contain the current face material
- "Hide all faces" - Hides all faces
- "Show selected faces" - Shows the selected faces (+ currently visible faces)
- "Hide selected faces" - Hides the selected faces

To show only the currently selected faces you can use the "Hide all faces" and "Show selected faces" menu items in succession.

The popup menu can also be used to select the current UV and the current face material name of the selected SDS object. The UV:s are listed first and the face material names last (there is a delimiter between the two groups). The SDS specific menu of the UV editor window allows you to detach materials from faces:

- Detach - detaches the current face material from the selected faces.
- Detach from all faces - detaches a face material from all faces. The face material is selected from the sub menu.

You can animate face uv coordinates by modifying them in animation recording mode.



UV editor's property window compass tab

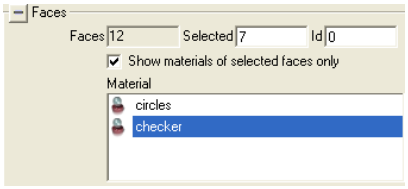
The Properties item in the UV View's pop-up menu allows you to open the property window for the UV View. Through this window you can control options like the contrast (weight) of the backdrop image, grid, rendering options and compass menus.

Note that you can record all options in this window to macros and bind those macros to desired keys, compass menus or icons for faster access.

Editing Face Materials through the Property Window

The object specific part (the Spec tab') of the property window shows you the materials assigned to the selected faces. Click the desired material to make it the *current face material*. Use drag & drop to change the order in which the materials are evaluated. The pop-up menu allows you to remove materials from faces.

The SDS property gadget has a checkbox 'Selected faces only' inside the Faces frame. If checked, the Material list works as before: shows the materials of the first selected face and its Detach popup menu removes the selected face material from the selected faces. If not checked, the Material list shows all face materials and Detach removes the selected face material from all faces.



In the Property window/Spec tab you can select the current face material, delete materials and change the order in which materials are applied

Map2Obj

This tool provides you with more control over assigning face specific materials.

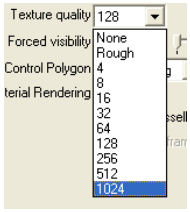
Map2Obj tool (located in the Materials tab) allows you assign materials to faces and define face specific UV coordinates using any mapping object, such as the parallel map. For example, you can use a parallel mapping to attach an 'eye' texture to the faces representing an eye of a head.

1. Select the desired SDS faces
2. Create a parallel mapping object (from any material)
3. Multiselect the SDS object and the parallel mapping object and activate the Map2Obj tool
4. Set Target to Selected Faces or All Faces, select the material to be assigned using the Destination Material field and click Accept.



Assign marble material to selected faces using the Map2Obj tool

Previewing Face Materials in OpenGL



Define texture quality through the Property window's Wire tab

To preview a single face material in OpenGL, open the property window, go to the Wire tab and set the Material Rendering option to Single. Specify also a suitable texture quality. Material Rendering=Full evaluates all mapped materials and is usually slower.

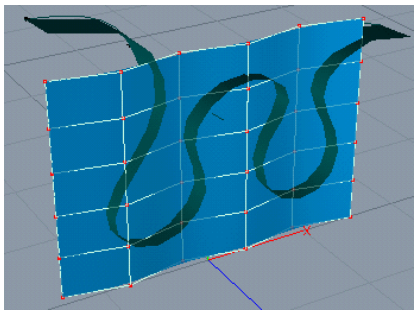
After the texture quality is specified, the system converts the current face material to a bit map image (texture map). You can find the generated images from the 'temp' folder. The image is named as 'materialname_texture.bmp'. Next time you attach the material to a face, OpenGL will be able to show the result instantly.

Using the Knife tools

The SDSPlus plugin must be installed to be able to use these tools. Two knife tools are available: the Object Knife Tool and the Freehand Knife Tool.

Object Knife Tool

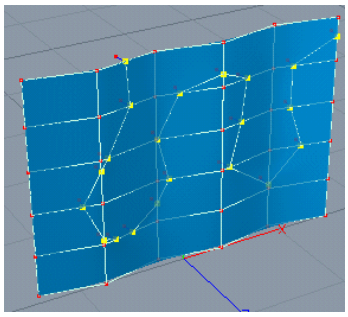
You can use the SDS Object Knife tool to create new points and edges to selected SDS objects. The surface of the last selected object is used as a knife to cut the selected SDS objects. New vertices are added to edges where the edges pass through the surface of the knife object.



A Nurbs surface (knife) passing through a rectangular mesh

You can find this tool from the SDS tab of the default working environment. The tool has the following options:

- Selected - if checked only the selected edges are affected
- Add edges - new edges are added between the new vertices
- Select - which faces to select. The SDS object must be in face selection mode. The values are:
  - No selection - does not change selection
  - Inside - select faces whose center point is inside the knife object
  - Outside - select faces whose center point is outside the knife object



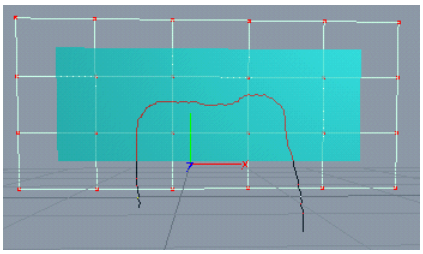
The knife tool applied

Freehand Knife tool

A curve or a line segment can be drawn defining a surface used as a knife. New vertices are added to edges where the edges hit the surface of the knife. This tool can be activated buy clicking the "Knife" button on the SDS control bar or by selecting the "Methods/Sub Division Surface/Knife" menu item.

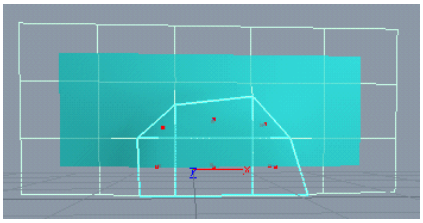
The tool has the following options:

- Selected - if checked only the selected edges are affected.
- Add edges - new edges are added between the new vertices.
- Freehand - if checked a freehand curve can be drawn. Otherwise a line segment can be drawn.

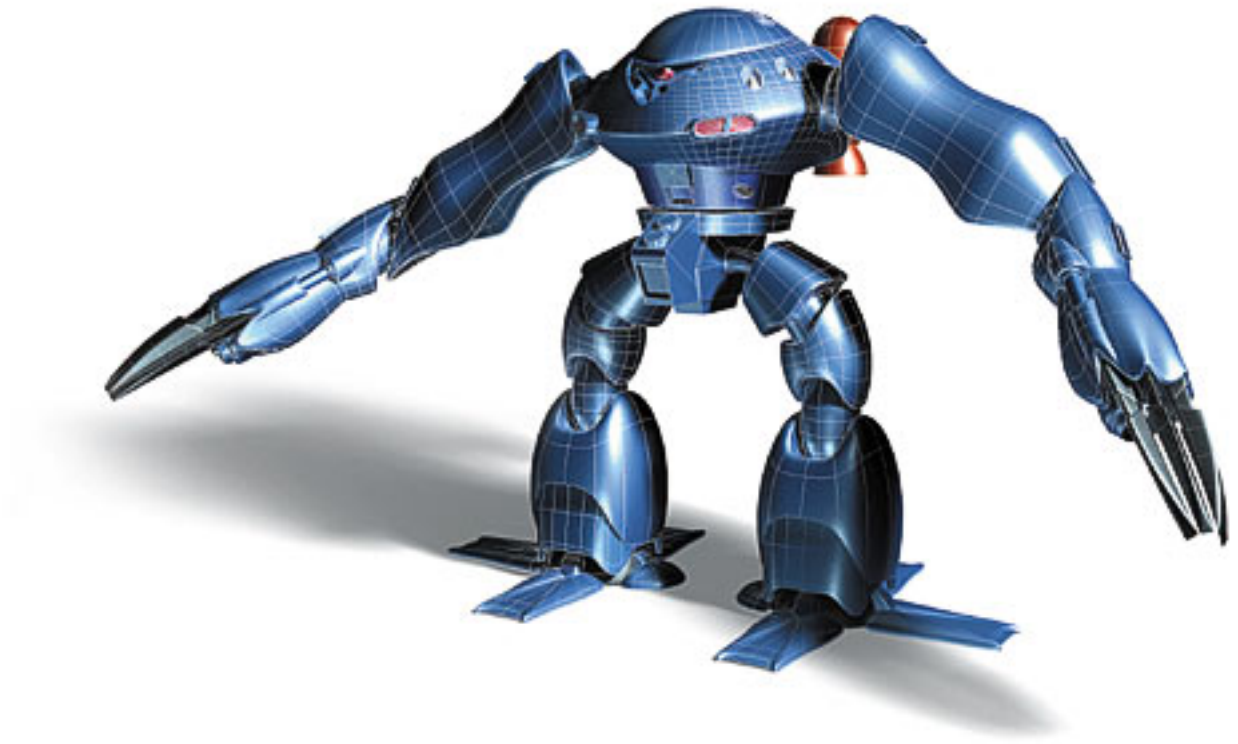


The Knife tool activated and freehand curve drawn over a SDS mesh

- Select - which faces to select. The SDS object must be in face selection mode. The values are:
  - No selection - does not change selection
  - Inside - select faces whose center point is inside the knife object
  - Outside - select faces whose center point is outside the knife object



Freehand knife applied



Chapter 5. NURBS Curves and Surfaces

NURBS Curves

NURBS modeling is often based on the idea that the user first creates one or more freeform curves and then applies appropriate surface construction tools to the curves.

In this tutorial we explore various NURBS curve manipulation tools.

To create a cubic NURBS curve:

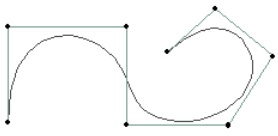
1. Click the NURBS/Curve tool.



The curve tool activated. The control bar shows the curve specific options.

As always, the control bar shows you the curve tool specific options. The default options are set for open cubic NURBS curves.

3. Click the left mouse button in the View window to enter a desired number of control points for the NURBS curve. If you click a new point in a wrong place, select Go Back from the view's popup menu to remove the last entered point. When done, select Accept from the tool control bar or from the view popup menu.



Create a cubic NURBS curve by entering a desired number of control points

Editing NURBS curves

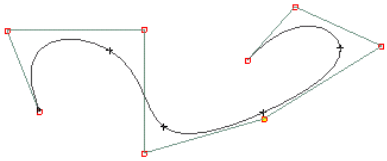
Similar to most other geometric objects, also NURBS curves specify object specific handles which can be accessed in the Edit mode.

For example, to change the position of the first control point:

1. Select the curve and enter the Edit mode (hit the Space bar, for example).

You can now see the control polygon of the curve.

2. Drag the first control point to move it.



Curve modified in the edit mode

You can also modify the curve using any transformation tool, such as the Move tool. First select some control points by using dragging or by clicking them while holding down the Shift key. Then use the move tool the usual way to move the selected points.

You can also modify the knot points (the actual curve points). Knot points are represented as small crosses on the curve. Just drag them to move them.

Interpolation curves

You can also create a NURBS curve by defining the actual curve points (the knot points). This is called 'interpolating' or 'inverse design'.

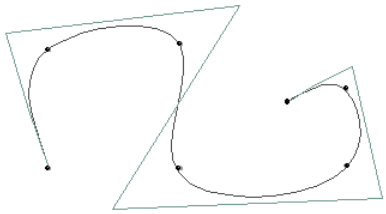
To create a curve which passes through the desired points:

1. Activate the curve tool and set the Interpolate check box.



Set the Interpolate option

2. Enter a number of points through the view window and select Accept.



A curve passing through the specified knot points

Creating freehand curves

The curve tool can be used for creating freehand curves. When using the Freehand option, you don't define curve by clicking the left mouse button. Instead, you drag with the mouse to define a freehand shape. The tool then creates a curve which represent the specified path with a minimal number of control points.

To create a freehand curve:

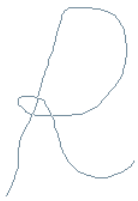
1. Activate the curve tool.

2. Set the Freehand option.



Freehand option set

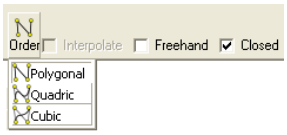
3. Press down the left mouse button in the view window and move the mouse to define a desired freehand shape. When done, release the mouse and the curve is created.



Hold down the left mouse button while moving the mouse to create a freehand curve

Controlling the order

The Order option controls whether the created curve is polygonal, quadric or cubic.



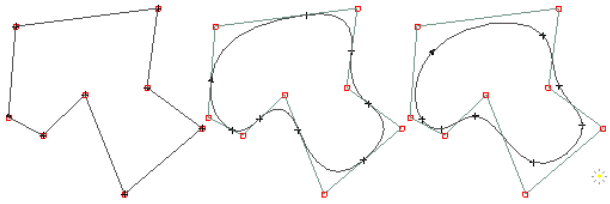
Select Polygonal order and check the Closed option

To create a closed polygonal curve:

1. Activate the curve tool.

2. Set Order to Polygonal and check the Closed option.

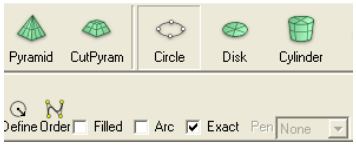
3. Click a desired number of points through the view window. Finish the curve by clicking the Accept button.



Polygonal, quadric and cubic NURBS curves

Creating circles and rectangles

The NURBS tab also has tools for creating circles and rectangles. These tools can be used exactly the same way as the Curve tool to create NURBS curves. In many cases you have to create circles or arcs. The circle tool can be used for this purpose.



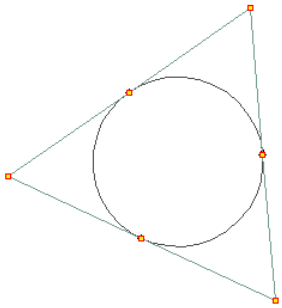
Circle tool selected with the Exact option set

For example, to create an exact NURBS circle:

1. Activate the NURBS/Circle tool.

2. Check the Exact check box. Make sure that the Filled option is not set.

3. Enter two points through the view window to define the center and the radius for the circle.



An exactly circular NURBS curve

You can also create non-exact circles. In this case, you can freely control the number of control points used in defining the circle. Such a circle is not mathematically exact but is often useful when creating various freeform objects.

To create a circle consisting of 8 control points:

1. Activate the NURBS/Circle tool.

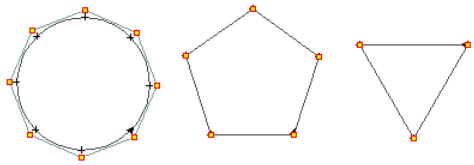
2. Define a desired number of control points to the Point field of the control bar.

3. Define the center and the radius in a view window.



Note

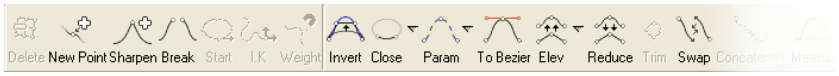
You can also create polygonal circles. For example, a triangle can be represented as a closed, polygonal NURBS curve consisting of three control points.



A cubic circle consisting of eight control points on the left. Two polygonal circles on the right.

Inserting points

When you select a NURBS curve, the tool control bar automatically shows the curve specific tools, which can be applied to the selected curves.



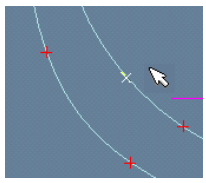
Control bar with the NURBS curve specific tools

To add a point to a curve:

1. Select a NURBS curve.

2. Click the New Point tool in the tool control bar.

3. Drag along the curve to find an appropriate position for the new point. The new point is indicated as a small cross. When you release the left mouse button, the new point is inserted to the curve.



Find a position for the point by dragging near the curve

Inserting new points to a curve does not change the shape of the curve. You can also create a new point by duplicating existing points. Holding down the 'Ctrl' key while dragging a knot point in the edit mode does this.

Creating sharp corners

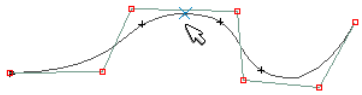
You can also add a new point to a curve so that the new point allows you to create a sharp corner to the curve. To add a sharp point to a curve:



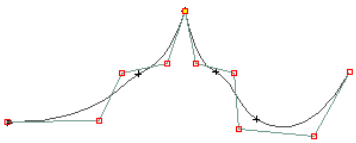
The Sharpen tool

1. Select the curve and activate the 'Sharpen' tool.

2. The Sharpen tool works exactly the same way as the Add Point tool: Move the mouse near the NURBS curve and drag with the left mouse button to find an appropriate position on the curve. When found, release the mouse button and the new point is inserted to the curve.



Drag along the curve to find a position for the new point



A Sharp point added and moved

The tool does not change the shape of the curve in any way. The nature of the inserted point is revealed when you modify the point.

3. Move the new point in the Edit mode. A sharp corner is formed.

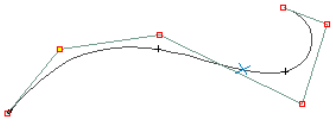
Breaking curves

The Break tool splits a curve into two parts.

The break point is defined the same way as the position of the new point in the 'Add Point' and 'Sharpen' tools:

1. Select the curve to be split and activate the 'Break' tool.

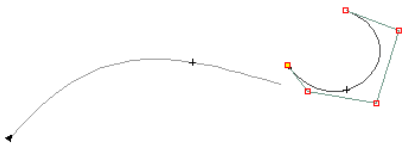
2. Drag along the curve to find a desired break point. Then release the mouse to accept the tool.



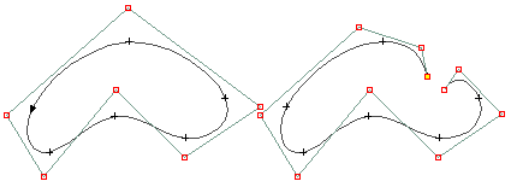
Drag along the curve to define the break point

The shapes of the resulting two curves represent together exactly the shape of the original curve.

You can apply this tool to a closed curve to open it from a desired point. Again, the shape of the curve remains unchanged.



A curve split into two parts. The first point of the tail part moved.



The original closed curve on the left. The Break tool applied and end points modified on the right.

Defining the start point of a closed curve

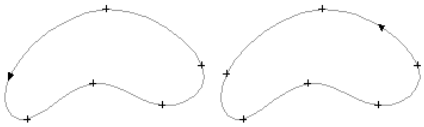
NURBS curves are 'parametric curves'. The beginning of a curve corresponds to the parameter value 0 and the end point of the curve corresponds to the parameter value of 1. However, a closed curve does not have an apparent beginning - any point could start the curve.

The 'Start' tool can be applied to translate the parameter space of a closed curve so that the given point becomes the start point of the curve. The start point is rendered as a small triangle. This can be relevant when using closed curves in path animations or as parameters for constructing surfaces.

The new 'start point' can be set to any position of the curve. The operation does not change the shape of the curve.

1. Select a closed curve and activate the 'Start' tool.

2. Drag the mouse along the curve. Release the left mouse button at a desired point.

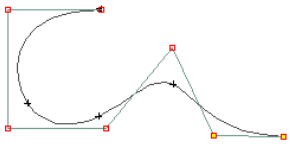


Two identical curves with different start points indicated by a small black triangle

Deleting control points

To delete one or more control points from a curve:

1. Select the curve and hit the space bar to enter the Edit mode. The curve now reveals its control polygon. Select the control points to be deleted.

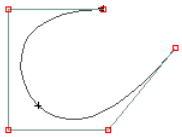


Two rightmost control points selected

3. Click the Delete button in the control bar.



Delete Control Points tool



The selected control points deleted

The shape of the remaining curve is changed by the operation. You can use the 'Break' tool to get rid of a part of the curve without affecting the shape.

Weights

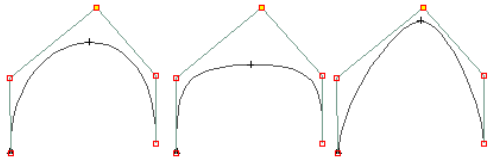
NURBS curves support a parameter called 'weight'.

The weight parameter allows you to control how strongly a control point affects the corresponding curve point (knot point). The higher the weight, the stronger the control point pulls the curve towards the control point.

You can modify the weight parameter interactively as follows:

1. In Edit mode, select the desired control point.

2. Activate the Weight tool from the control bar and click the mouse in the view window. By moving the mouse up/down you can increase/decrease the weight of the selected control points. Click the mouse again to accept the tool.



Three NURBS curves with different weights at the topmost control point



Note

You can also modify the weight of the selected control points by dragging them while holding down the Alt key.

Degree elevation and reduction

The NURBS Curve tool can create polygonal, quadric and cubic nurbs curves using the selected Order parameter.

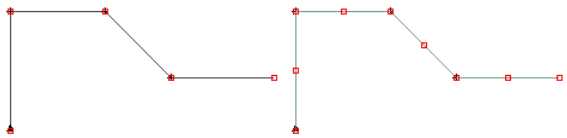
The order of a polygonal curve is two. Mathematically speaking, such a curve is of first degree. Second degree curves are called 'quadrics' and they can represent accurately circles and ellipse arcs, for example. Cubic curves are often a very good compromise between smoothness and performance. However, even higher order curves can be created. The higher the degree, the smoother the curve.

The Elevate and Reduce tools modify the degree of a curve.

Assume that you have a polygonal curve and you need to bend its tail smoothly. A polygonal curve does not allow smooth bending. Therefore:

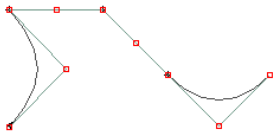
1. Create a polygonal curve.

2. Select the curve and click the Elevate button on the control bar.

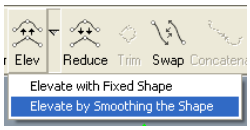


The original polygonal curve on the left. The degree elevated quadric curve on the right.

As you can see, the degree elevation didn't change the shape of the curve. This is possible because quadric NURBS curves are flexible enough to represent also polygonal shapes. To see that the curve is really quadric, edit the last control points to bend the tail.



Two control points moved

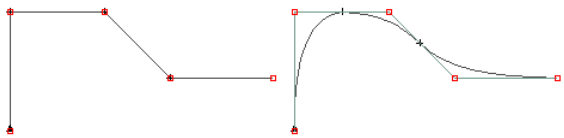


Degree elevate and smooth

You can also elevate the degree of a curve so that the increased order is used to smooth the curve. The Elevate tool includes a pop-up menu, which selects the elevation method. Click the small arrow associated with the Elevate button to open the menu.

High order NURBS curves are useful when you need extra smooth and continuous curves for motion paths. High level of smoothness is important in camera fly animations, for example.

To reduce the degree of a curve, just click the Reduce tool. Note that degree reduction always changes the shape of the curve.



The original polygonal curve on the left, Elevate by Smoothing applied on the right

Concatenating curves

As demonstrated above, NURBS curves can represent both polygonal and smooth shapes. How about creating a single curve, which contains both polygonal and smooth sections?



Concatenate multiple curves to one

An easy way to achieve this is by using the Concatenate tool.

1. Create a cubic curve.

2. Create a polygonal curve, which starts exactly from the end point of the cubic curve.



Note

Use dragging to snap the position the first point of the polygonal curve exactly over the last point of the cubic curve.

3. Select both curves (using the Shift key) and click the Concatenate tool to get single curve with cubic and polygonal curve segment.



Concatenating two curves of different order



Note

If the end points of the curves are displaced, the tool creates one additional curve segment to fix the displacement. If the end points overlap, the result matches exactly the shape of the original parameter curves.

You can select the curves for concatenation in any order. The tool automatically finds correct order by studying the distances between the end points of the selected curves.

Concatenate is a very powerful curve construction tool, because it can join curves with different parametrization and order to one single curve.

Parametrization

Parametrization of NURBS curve is a rather mathematical issue and there is no easy way to explain it. A great deal of the power behind NURBS curves relies on the possibility to use a suitable non-uniform parametrization. Fortunately, in most cases, you don't have to worry about the parametrization issues.

The Param tool changes curve parametrization. The following parametrization methods are supported by the tool:

- Uniform.
- Chord Length
- Centripetal

Typically, Chord Length or Centripetal parametrization give the smoothest curve shape (and most constant speed when used as an animation path). However, sometimes you may want to use also Uniform parametrization.



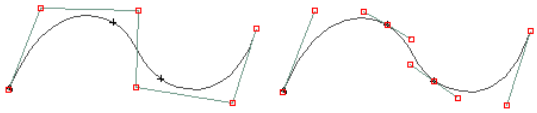
Note

Changing the parametrization always affects the shape of the curve. In fact, many of the tools we have demonstrated above modify the curve parametrization to achieve the desired effect. For example, the Break tool increases so-called 'knot multiplicity' so that a sharp corner can be represented. By applying the Param tool, you lose the custom parametrization needed for sharp corners. The result is a curve, which smoothly approximates its control points.

Bezier curves

Thanks to non-uniform parametrization, NURBS curves are powerful enough to represent also Bezier curves. However, this parametrization technique can only be used if the number of control points is appropriate. Therefore, it is not available as an option for the Param tool.

Instead, a special Bezier tool is included, allowing you to convert any NURBS curve to the Bezier form.



A standard cubic NURBS curve with a chord length parametrization on the left. The Bezier tool applied on the right.

The Bezier form provides you with a great deal of power over the shape of the curve. However, it also allows you to define curves, which are not as smooth as curves with uniform, chord length or centripetal parametrization. This can be a significant disadvantage when using curves to construct surfaces or when using them as motion paths in camera animations.

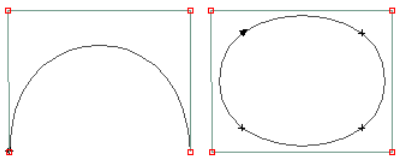
Opening/closing curves

You can close an open curve and open a closed curve by clicking the Close tool. If you have selected various curves and some of them are open, some closed, the pop-up menu associated with the Close button can be used to select whether all the curves should be opened or closed.

There are two ways to open curves: exact and approximate.

When using the exact method, the shape of the curve is not changed. The curve is broken from its start point and you can see that it is really open only by displacing its end points.

The second method simply takes the existing control polygon to define a new open curve. The shape of the curve is changed dramatically at the ends.



An open curve on the left. The result after applying Close on the right.



Note

If you have used Sharpen, Bezier or other tools which create a custom parametrization, the Close tool may introduce a sharp point where the curve is closed. You can fix the problem by changing the parametrization to uniform, centripetal or chord length.

Changing the direction of a curve

The Swap tool inverts the direction of a curve. The direction of a curve is significant when using it as a motion path or as a parameter for various surface construction tools. The direction of a curve is indicated by the small black triangle.

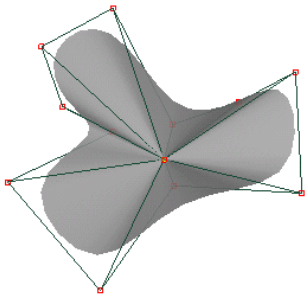
Just select desired curves and click Swap button on the control bar to change their direction.

Inverting curves

The Invert tool changes the curve so that it passes through its original control points. The result is the same as if you created the curve using the Interpolate option in the first place.

Using the Extrude option

Another way of creating curves is using the curve tool that is in the Tools/Unified menu. This curve tool also supports the Extrude option. When the Extrude option is set, the curve tool creates various NURBS surfaces instead of curves. For example, to create a conical surface with a freeform cross section:



A conical surface with a freeform cross section

1. Activate the Unified/Curve tool.

2. Set the Extrude option.

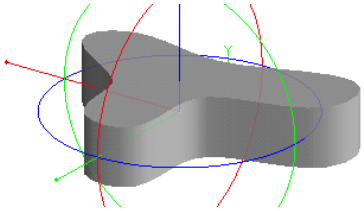
3. Enter a number of control points to define a cross section for the extruded surface. When done, click Accept. You can now define the position for the apex. Click the mouse in the desired point and select Accept.

If you want to create a cylinder with a freeform cross section:

1. Activate the Unified/Curve tool. Set the Closed option from the control bar.

2. Set Extrude. You may also set Filled to get lids for both ends of the cylinder.

3. Enter the cross section points and select Accept. The curve tool starts to outline the position for the apex. Don't enter the apex point. Instead, select Accept again.



A cylindrical object created by the curve tool with Extrude and Filled options

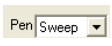
Using the Pen option

The NURBS Curve tool allows you to construct surfaces interactively. This can be achieved using the Pen option. By default, the pen is just a single point - sweeping it in 3D space creates a curve. When the Pen option is activated, the selected object is taken as a 'pen'. For example, by using a circular curve as a pen for the NURBS Curve tool, one can create circular tubes.

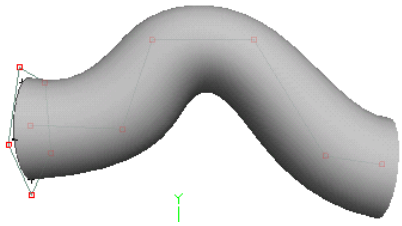
Let's try this:

1. Create a NURBS circle using the NURBS Circle tool from the side view.

2. Make sure the circle is selected and activate the NURBS Curve tool. Then set the Pen option to Sweep.



Set Pen to Sweep



Create a tube by using a NURBS circle as a pen for the curve tool

3. In the front view, define a desired profile curve for the tube. Make sure you click the first profile point over the selected circle. A circular tube is shown while you define the profile curve.

You can create open surfaces by using an open curve as a pen. If you use a polygonal curve as a pen for a cubic curve, the result is a surface which is polygonal in one direction and cubic in the other direction.

The Rotate pen rotates the drawn curve about the selected axis object. For example, to create a banana:

1. Create a slightly bent axis curve, which travels in the middle of the banana.

2. Select the axis curve, start the NURBS Curve tool, set Pen to Rotate and enter a profile curve for the banana.

Constructing surfaces is described in more detail in other NURBS modeling tutorials.

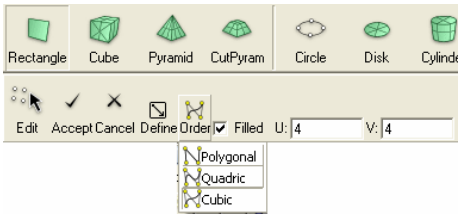
NURBS Surfaces

Many of the direct curve manipulation tools, such as Break and Sharpen, can also be applied to surfaces. The only difference is that surfaces have two parameter directions (called u and v) in which they can be manipulated.

In this tutorial, we will demonstrate these direct NURBS mesh manipulation tools.

The NURBS tab contains tools to create basic NURBS surfaces with freeform, circular and rectangular shapes.

Let's create a rectangular cubic NURBS mesh, which consists of 4 x 4 control points:



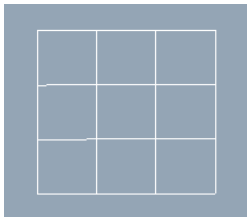
NURBS Rectangle tool activated. Options for creating a 4x4 cubic mesh.

1. Activate the NURBS/Rectangle tool.

As always, the control bar shows you the tool specific options.

Set Order to cubic, make sure Filled is checked and specify the control point count 4x4 through the U and V fields.

2. Then enter two points through a view window to define a geometry for the rectangle.



A rectangular NURBS mesh consisting of 4x4 control points

Curve handles

To switch to the curve editing mode:

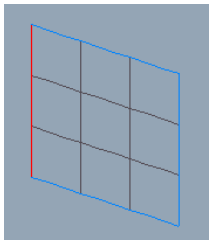
1. Select the mesh. As usual, the control bar shows you the tools, which can be applied to the selected mesh.



NURBS mesh specific tools

2. In the control bar, set the Edit control to Curves. Just click on the Edit icon and select Curve from the drop down menu. The mesh now shows you two vertical and two horizontal curve handles. When you move the mouse over a curve handle, the color of the curve is changed. As usual, you can move any curve simply by dragging it with the mouse.

You can also multi select any number of curves and apply any transformation or deformation tool to them.



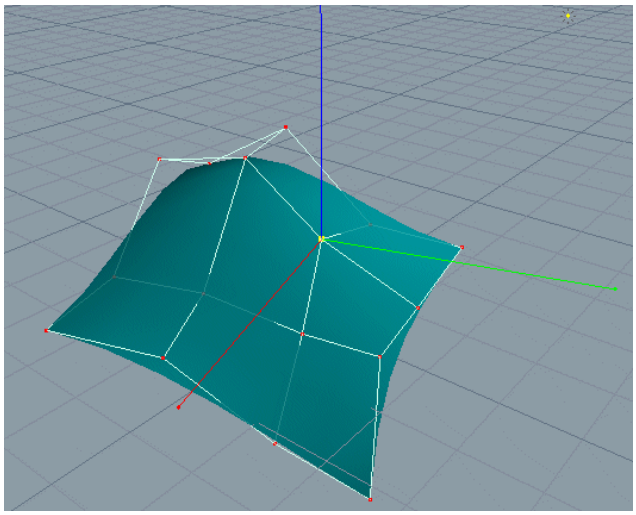
The left vertical curve selected and moved upwards

Control point handles

To enter the control point editing mode, set the Edit control in the control bar to Point to access the control point handles.

Hint: use Compass menus to switch between different editing modes easily.

Just drag the control points, using the modification handles with the mouse to move them.



Control point handles shown.

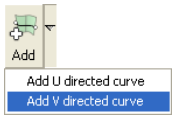
Adding new curves

Just like you can add a new point to a curve, you can add a new curve to a mesh.

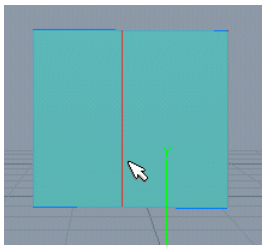
1. Use the NURBS/Rectangle tool to create a planar cubic NURBS mesh consisting of 4x4 control points.

2. Select the Add V Directed Curve pop-up menu item of the Add tool.

Hint: you can also use the compass menu to activate the tool.

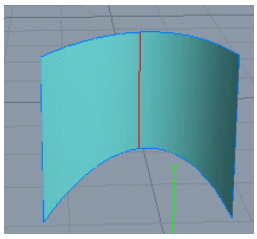


Activate the Add Curve tool with the V Directed Curve option



Activate the Add tool and LMB drag over the mesh to find a desired position for the new curve

3. Now press down the left mouse button over the mesh. Move the mouse while still holding down the left mouse button. The tool renders the new curve over the mesh. When a desired position is found, just release the mouse.



The new curve moved and scaled

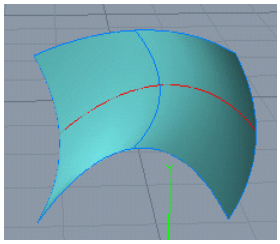


Note

The shape of the mesh is not affected by the new curve. So it does not make any sense to add new curves without modifying them.

Similarly, you can add a new horizontal curve to the mesh. Just select Add U Directed Curve from the pop-up menu of the Add tool and drag over the mesh to define the position for the curve.

Most of the direct mesh manipulation tools work this way.



A new U directed curve added and moved

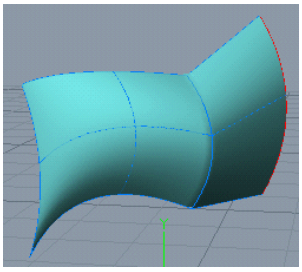
Creating sharp curves

The Sharpen tool works similar to the Add tool. The only difference is that the inserted curve breaks the continuity of the mesh so that a sharp edge can be formed.

Let's imagine we need to add a sharp V directed curve to the mesh we created above.

1. Select the Add V Directed Curve of the Sharpen tool and drag over the mesh to specify a position for the curve. The workflow is exactly the same as demonstrated in the previous example.

The Sharpen tool doesn't change the shape of the mesh. However, the nature of the new inserted curve is revealed as soon as you modify the mesh. For example, move the rightmost vertical curve upwards a bit.



Sharp v-directed curve added and the rightmost curve moved upwards

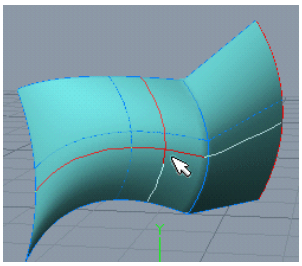
Breaking meshes

The Break tool breaks the mesh through the specified curve. Again, the workflow is similar to the previous Add and Sharpen tools.

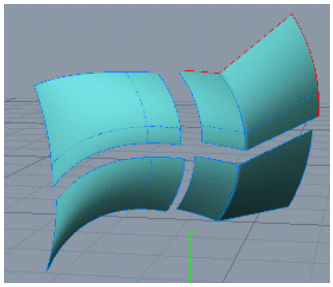
Let's split the mesh we created in the previous example into four parts:

1. Click the Break tool button.

2. Drag over the mesh to define the positions for the U- and V-directed curves.



The break tool activated with both U and V options

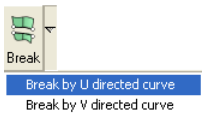


Pieces moved apart

As you can see, two perpendicular curves passing through the specified point are rendered over the mesh. The reason for this is that clicking the Break button selects both U and V options. This is also true for Add and Sharpen tools.

When a desired position for the curves is found, just release the mesh and the mesh is split into four parts by the curves.

If the mesh is closed in the given direction, the specified curve just opens the mesh through the specified curve. Let's demonstrate this by applying the break tool to a NURBS sphere. The surface of a sphere is closed in its U direction but open in its V direction.



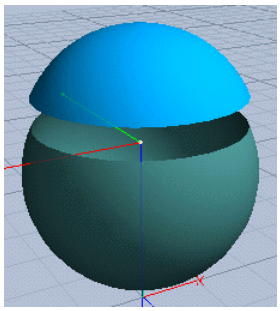
Break by U Curve

1. Switch to the top view and create a cubic NURBS sphere by using the NURBS/ Sphere tool.

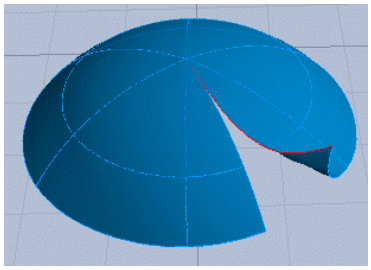
2. Switch to the front view and select Break by U Directed Curve pop-up menu item of the Break tool.

3. Drag over the sphere to find a desired break curve. When found, release the mouse to accept the tool.

Because the U directed curve forms a closed loop (the mesh is closed in its U direction), the tool split the surface into two separate parts. Correspondingly, the result is two separate NURBS meshes.



North pole cut off



Hemisphere broken by a v-directed curve and edited

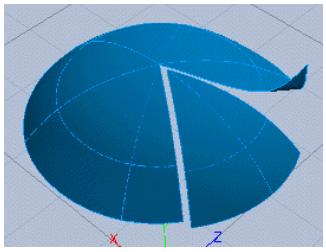
So, what happens if we break the mesh in its V direction? In that direction the surface is open: All V directed curves have end points at the poles of the sphere.

4. Select the north-pole mesh and activate Break by V Directed Curve. Then drag the mouse over the hemisphere to find a desired position for the break curve. When found, release the mouse.

As you can see, new objects were not created. Also, the shape of the mesh remains exactly the same. However, the mesh was opened through the specified curve, as you can see by modifying the mesh.

Now the semi sphere is open in the both parameter directions. Let's apply Break by V Directed Curve again.

Now the result is two separate meshes.



Defining start point of a closed mesh

Just like you can redefine the start point for a closed curve, you can redefine the start curve for a closed surface. Again, the only difference to the curve start tool is that now there are two parameter directions U and V in which the start curve can be defined.

Weights

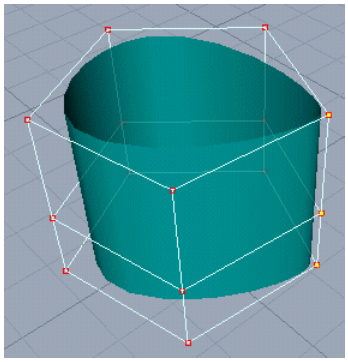
NURBS meshes support a parameter called Weight.

The weight parameter controls how strongly the control point in question affects the corresponding surface point (knot point). The higher the weight, the stronger the control point pulls the surface towards the control point.

You can modify the weight parameter interactively as follows:

1. In Edit mode, select the desired control points.

2. Select the Weight tool and click the mouse in a view window. By moving the mouse up/down you can increase/decrease the weight of the selected control points. Click the mouse again to accept the tool.



A high weight value set to three selected control points



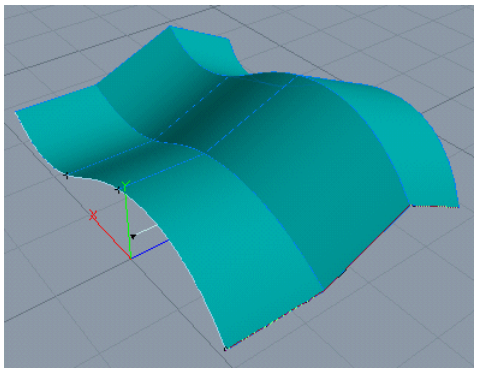
Note

You can also modify the weight of the selected control points by dragging them while holding down the Alt key.

Degree elevation/reduction

The *Degree Elevation* and *Degree Reduction* tools modify the order (degree) of the mesh. We already demonstrated the effect of degree elevation/reduction in the context of NURBS curves. The only difference to NURBS curves is that NURBS meshes have two directions in which the order can be controlled independently.

Polygonal, quadric and cubic meshes are sufficient for most practical applications. Surfaces with higher degree are sometimes needed when an extremely high surface smoothness and rendering quality is desired.

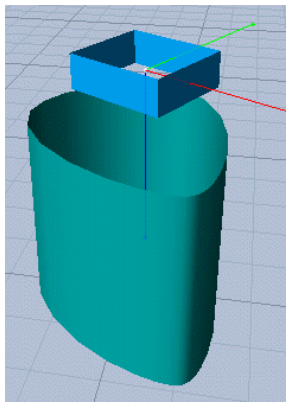


A mesh, which is cubic in u-direction and polygonal in v-direction

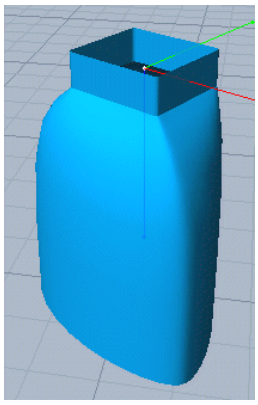
Concatenating meshes

The Concatenate tool concatenates two meshes with each other provided that the meshes have an edge i.e. they are open at least in one parameter direction. The tool concatenates the two nearest edges.

To use the concatenate tool, move the meshes to be concatenated so that the edges, which should be connected with each other, are placed side by side. The tool finds the nearest edges and connects them.



Original meshes

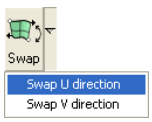


Concatenate tool applied

Swap

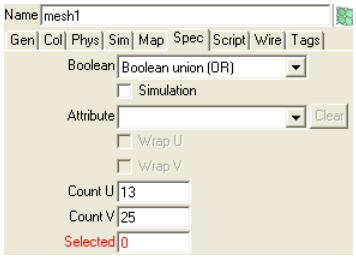
Sometimes you have to swap the direction of a mesh. For example, it may be necessary to invert the direction of a mesh to get it concatenated properly to another mesh.

The Swap tool swaps the direction in both u- and v-directions. To swap the direction in one direction only, select the appropriate pop-up menu item from the tool button.



Controlling the resolution of a mesh

Let's imagine you have a mesh and you need to double the number of control points without changing the shape of the mesh. Editing the resolution attributes in the property window/spec tab does this. For example, you can double the value of the Count V field.



Edit the Count U/V fields to control the resolution of the mesh



Note

If the resolution is increased, the shape of the mesh is not changed. Decreasing the resolution usually changes the shape.

Opening/closing meshes

You can close an open mesh and open any closed mesh by clicking the Close tool. The tool opens the mesh from its start curve.

If you have a number of meshes and you want to make them all closed, select the Close pop-up menu item associated with the Close tool.



Note

You can also use the Break tool to open a closed mesh from an arbitrary position.

Inverting meshes

The Invert tool allows you to change the mesh so that it will pass through its original control points. Just click the tool button to apply the tool.

For example: if you import a polygonal mesh and want to convert it to a smooth cubic mesh, inverting the converted mesh compensates for the shrinking effect of degree elevation by smoothing.

Editing the current selection

The arrow keys (Up, Down, Left, Right) can be used for changing handle selection (curves/control points) in the Edit mode. The left and right arrow keys can be used to move the current selection in the U parameter direction. Correspondingly, the up and down keys allow you to move the selection in the V direction. This is very handy when modifying complex meshes. For example, to expand the current selection in the U direction, hold down the Shift key and hit the left and/or right arrow keys to select more curves.

Converting other objects to NURBS mesh form

The To Nurbs tool converts geometric objects to NURBS form by approximating them with a set of NURBS meshes. For example, to convert an analytic cube and sphere to meshes:

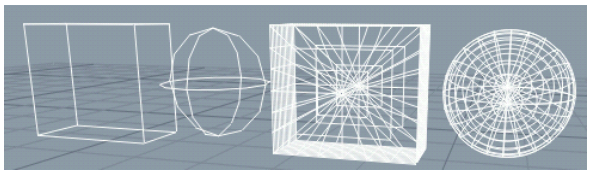


To NURBS tool

1. Multi select the cube and the sphere.

2. Activate the To Nurbs tool from the NURBS tool tab.

3. Specify the approximation density values Density U and Density V in the tool control bar. The default values U=1, V=1 define the minimum density. Set U=2, V=2. These values are sufficient for most cases. Click Accept to finish the conversion.



An analytic sphere and a cube converted to NURBS meshes

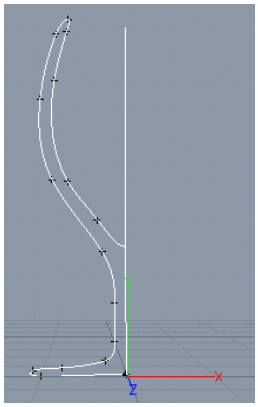
Rotational Surfaces

Objects with a circular cross section can be constructed from two NURBS curves: an axis of rotation and a profile curve. We use the classical example of creating a wine glass for the demonstration.

1. Use the NURBS/Curve tool to create a vertical line in the view window. The length of the line does not matter, except that a long line is helpful when drawing the glass profile. Rename it to 'axis curve'.

2. Create another curve representing a suitable wine glass profile. Rename it to 'profile curve'.

3. Multi-select the curves using the Shift key, the axis curve first, and the profile curve next.



An axis curve and a profile curve

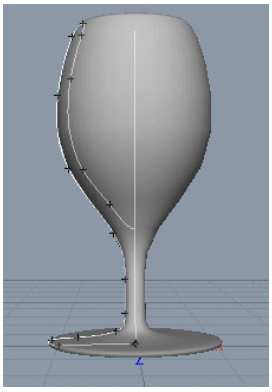
4. The control bar now shows you the tools, which can be applied to the selected curves. Select the Rotate tool.

 *Rotate tool*

The Rotate tool takes the first selected curve as an axis of rotation and the second curve as a profile curve.

If you accidentally selected the curves in the wrong order (in which case the tool rotates the axis about the profile curve), simply select the curves in the correct order again. There is no need to restart the tool.

5. The default options are fine for this example - just click 'Accept'.



A wine glass

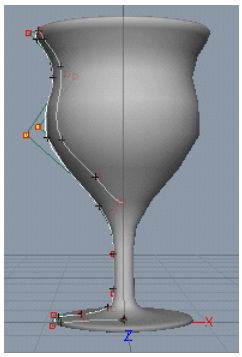
Editing the construction curves

NURBS mesh construction tools allow you to access the original construction curves and edit them in order to modify the shape of the mesh. For example, let's assume that the profile of the glass is not exactly what you expected and you need to make the wine glass slimmer.



The hierarchical structure of the wine glass

1. In the select window, click the '+' handle of the created NURBS mesh object. You can now see that the NURBS mesh includes two sub objects - the curves you used to create the wine glass in the first place.



Control points of the profile curve modified

2. Select the profile curve and enter the 'Edit' mode (press Space bar) to access the control points of the curve. Modify the profile curve. The shape of the wine glass is updated with the curve.

Note

In addition to moving points, you can add new points, delete existing points, add sharp edges, and so on. You can even replace the entire profile curve with another curve to rebuild the shape of the glass. If you model wine glasses a lot, you may build a library of profile curves. Just drag & drop the desired profile curve inside a wine glass mesh to get a desired wine glass constructed!

Note

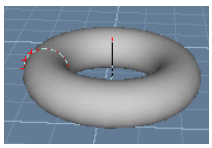
You might want to control the 'Forced visibility' attribute of the NURBS curves when working with the shaded OpenGL mode. This attribute makes the curves visible even if there they are hidden by the actual surface of the wine glass. The higher the 'forced visibility', the better they can be seen. You can access this option in the property window's 'Wire' tab.

Tutorial project: 'tutorprojects\modeling\nurbs\rotate\wine glass'

Torus

A torus surface can be created by rotating a NURBS circle about an axis.

Tutorial project: 'tutorprojects\modeling\nurbs\torus.r3d'

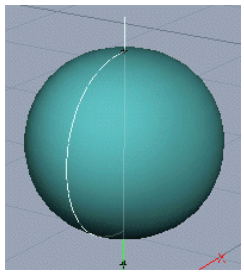


Rotate a circular curve about an axis to get a torus

Sphere

Rotating a semi circle about an axis creates spheres and ellipsoids.

Tutorial project: 'tutorprojects\modeling\nurbs\rotate\sphere'



Rotate a semi circle about an axis to build a sphere

Interactive Method

The Nurbs Curve tool's Pen option allows you to create rotated surfaces interactively.

1. Create or select a NURBS curve representing an axis. Note: the axis curve does not have to be a straight line. Any curve will do.

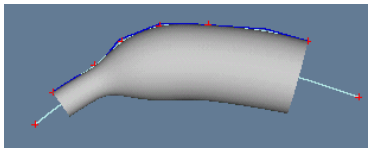


Set the Pen option to Rotate

2. Activate the NURBS/Curve tool and set the 'Pen' option to 'Rotate'.

3. Enter points for the curve as usual. New curve points are rotated about the selected 'axis' curve while you enter them.

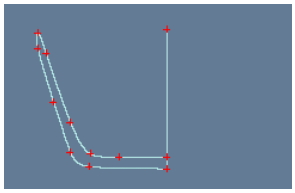
Tutorial project: 'tutorprojects\modeling\nurbs\rotate\banana'



Create a banana with the Rotate pen

Scale curve

If only two curves are selected, the rotation tool rotates the second curve about the first curve. A third curve can be selected to control the properties of the mesh in its 'v' direction.

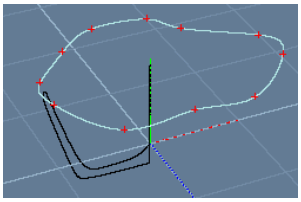


A side view of the axis and the profile

For example, to create a vase with a non-circular cross section:

1. Create an axis and a profile curve from the side view.

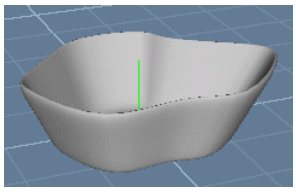
2. Switch to the top view and create a closed scaling curve.



The axis, the profile and the scale curves created

3. Select the axis, profile and scale curves (in this order) and click the Rotate tool in the control bar. Then click 'Accept' to finish the mesh construction.

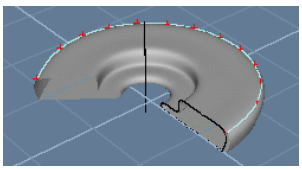
Tutorial project: 'tutorprojects\modeling\nurbs\rotate\vase'



A non-circular vase

If the scale curve is open, also the mesh is open in the corresponding direction. By using a 'helix' for the scale curve, one can create screws and other objects with a spiral shape.

Tutorial project: 'tutorprojects\modeling\nurbs\rotate\open scale curve'

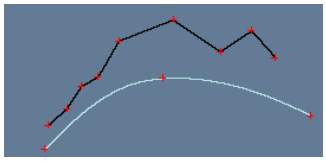


A circle arc used as a scale curve

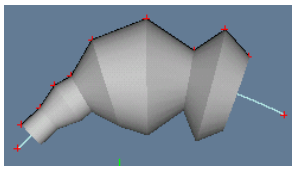
Controlling the order of the surface

Let's create another banana.

1. Create two NURBS curves: a slightly bent axis curve and a profile curve. However, this time, make the profile curve polygonal by setting the 'Order' option of the Nurbs Curve tool to 'Polygonal'.



Rotate a polygonal profile curve



Polygonal curve rotated

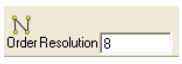
2. Select both curves: first the axis curve and then the polygonal profile curve. Activate and accept the Rotate tool.

The result is a mesh, which is polygonal in the 'v' direction and cubic in the 'U' direction.

Tutorial project: 'tutorprojects\modeling\nurbs\rotate\polygonal profile'

We can also create a banana, which is polygonal in its closed direction but smooth in the axis direction. The Rotate tool provides the necessary controls for this.

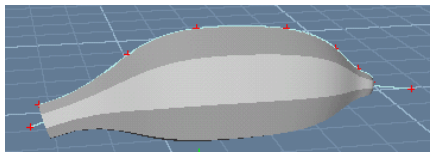
1. Create two smooth cubic NURBS curves: a bent axis and a profile for the banana.



Rotation controls, which determine mesh properties in the direction of rotation

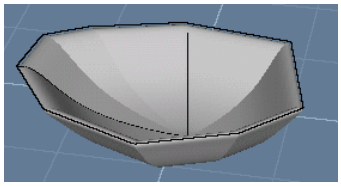
2. Select the curves and activate the 'Rotate' construction tool as shown above. The control bar now shows you the rotation specific options.

3. Set Order to Polygonal and click Accept.



Polygonal rotation

When using a 'scale' curve, the order of the mesh is fetched from it. For example, the following image demonstrates a combination of a cubic profile curve and a polygonal scale curve.



A cubic profile and a polygonal scale curve. The mesh is polygonal in the direction of the scale curve.

Sweep and Bi-Rail Tools

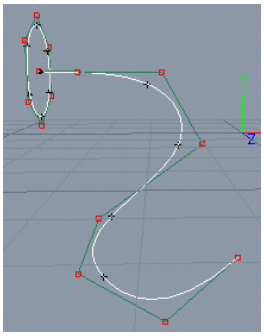
The Sweep tool constructs a surface by moving a cross section curve along one or two rail curves.

An 'S' shaped tube can be created by sweeping a circular cross section curve along an 'S' shaped rail curve:

1. Switch to the front view and create a closed NURBS curve.

2. Switch to the side view and create an 'S' shaped open NURBS curve which starts from the center of the first curve.

3. Select the cross section curve and the rail curve (in this order). As always, the control bar shows you the tools, which can be applied to the selected curves.



A closed cross section curve and an open rail curve

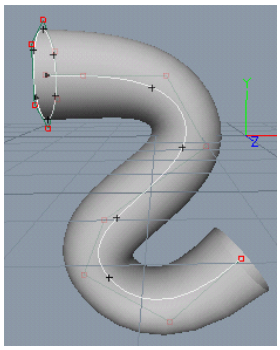
4. Click the Sweep tool.

There are two ways to sweep the circular cross section curve on the rail curve.

- Orthogonal sweep. The direction of the cross section curve is rotated with the curvature of the rail.
- Coplanar sweep. In this method the direction of the cross section circle remains the same.

Let's use the orthogonal sweep, so set this option in the control bar. Then click 'Accept'. Leaving the 'orthogonal' option unchecked in the control bar, results in a coplanar sweep.

Tutorial project: 'tutorprojects\modeling\nurbs\sweep\tube'



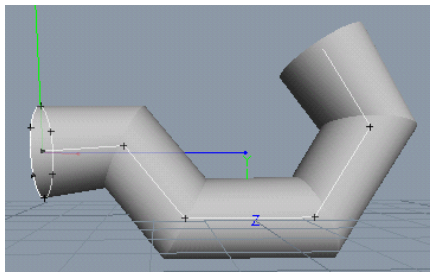
A tube created by sweeping a closed curve along an S-shaped open curve

Controlling the order of the surface

Let's assume you want to create a circular tube, which consists of straight cylinders. This can be constructed by sweeping a smooth circular cross section curve along a polygonal rail curve.

To create such a tube, just replace the sweeping curve of the previous example with a polygonal curve.

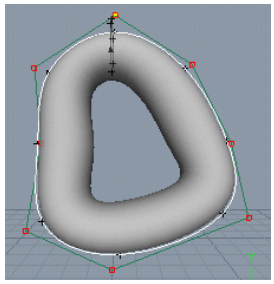
The curves, from which the surface is constructed, determine the mesh properties. If you want to create a completely polygonal NURBS surface, use two polygonal curves.



A cubic circle swept along a polygonal rail

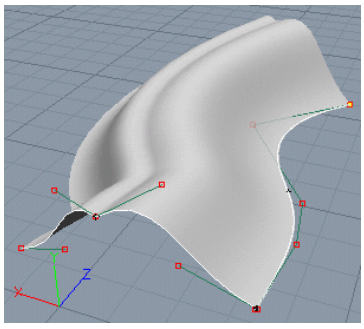
Tutorial project: 'tutorprojects\modeling\nurbs\sweep\polygonal rail'

In addition to Order, also the Open/Closed attributes of the mesh are determined by the corresponding state of the construction curves. A torus surface is closed in both directions. Such a surface can be swept by using two closed curves. Remember to start drawing the rail curve from the middle of the cross section curve.



A torus surface created by sweeping a circular cross section curve along a closed rail curve

The weight and parametrization of the construction curves control parametrization and weights of the mesh. For example, if you increase the weight of a control point, the weight of the corresponding 'isoparametric curve' is changed. Or, if you use a Bezier curve as a rail, also the surface will be Bezier-parametrized in the corresponding direction.

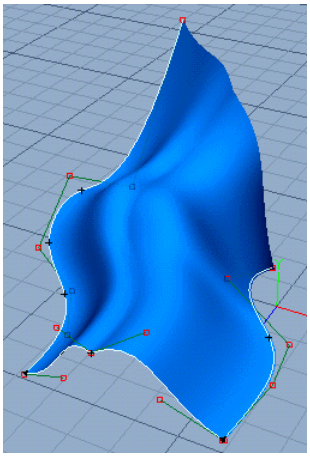


A Bezier curve used as a cross section curve

Bi-rail

The Sweep tool supports also two rail curves. Bi-rail sweeping activates automatically when you select three parameter curves for the tool. The first selected curve is taken as a cross section curve and the remaining two curves are used as rails to move, rotate and scale the cross section curve.

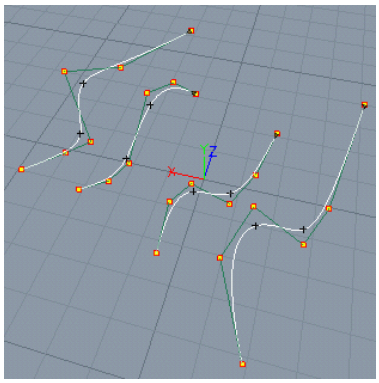
Tutorial project: 'tutorprojects\modeling\nurbs\sweep\bi-rail'



The bi-rail tool: a Bezier curve swept by two rail curves

Cross Section Surfaces

The Cross Section tool constructs a surface, which passes through a given set of cross section curves. To create a cross section surface:

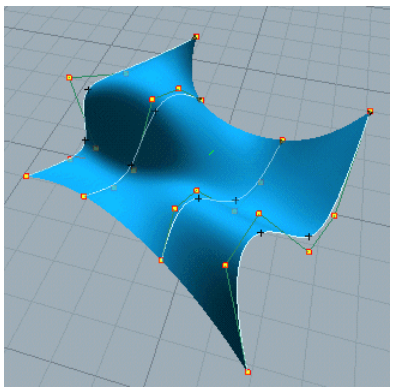


Four cross section curves

1. Create a set of NURBS curves.

2. Multi select the curves in the order in which the surface should interpolate through the curves. The control bar now shows you NURBS specific construction tools. Click the X-Sect tool.

 *Cross Section tool*



A NURBS surface created through the selected curves

You can now see two cross section tool specific options in the control bar: Closed and Order. We will create a cubic mesh through the curves. Therefore, set the Order to Cubic and click Accept to create a surface.

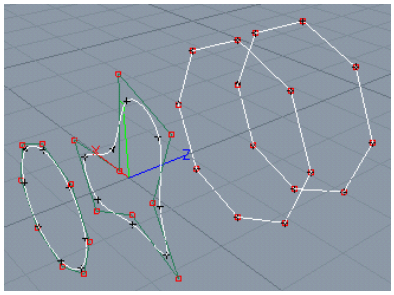
Tutorial project: 'tutorprojects\modeling\nurbs\cross_section\cross_sections'

The cross section tool can handle curves of any order. Some of the curves can be polygonal and some cubic, for example.

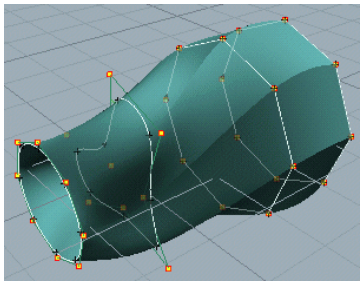
Let's create a tube, which starts with a round smooth shape and ends to a polygonal form.

1. Create two closed cubic NURBS curves and two closed polygonal curves, as shown in the next picture.

2. Select the curves from left to right (in the order in which you want to connect them). Activate the XSect tool, set Order to Cubic and click Accept.



Four closed curves defining the cross sections of the tube



A tube created through four closed curves with different orders

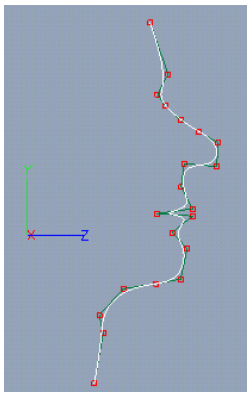
Creating a face from cross section curves

This tutorial demonstrates how to create a human face using the cross section tool.

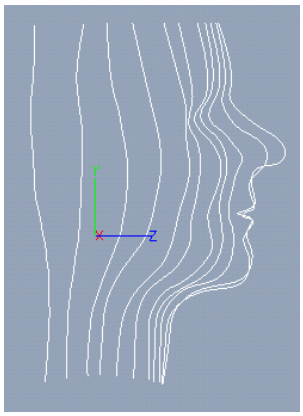
1. Create a curve representing the profile of a face.

2. Duplicate the profile curve and modify it to get the next 'slice' of a face. Continue this way until you have the necessary cross section curves for one half of a face.

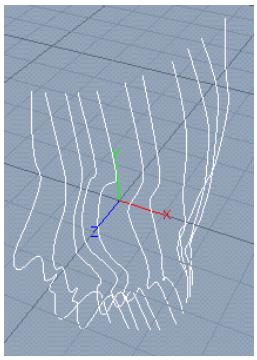
3. Now displace the curves by rotating them from the top view. You can also single point edit the curves from the front view to get non-planar curves.



A NURBS curve representing the profile of a face



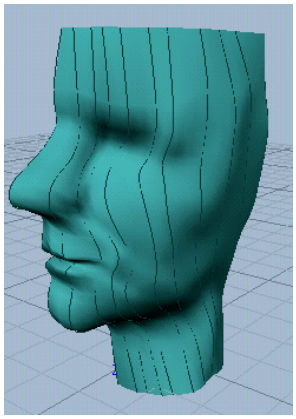
Cross section curves for one side of the face



Face curves seen from the top

4. Multi select the curves in the correct order and apply the XSect tool. Set the Order option to Cubic and click Accept.

Tutorial project: 'tutorprojects\modeling\nurbs\cross_section\semi head'



One half of a face defined by a number of vertical cross section curves

Editing construction curves

As usual, the original construction curves are placed inside the mesh and you can modify them in order to control the shape of the mesh.

If you delete one of the construction curves, the corresponding curve is also removed from the actual mesh. For example, to make the head we created in the previous example complete, duplicate and mirror the cross section curves and then drop the duplicates to the beginning of the construction history in inverted order (the order of the doubled curve set must be correct).

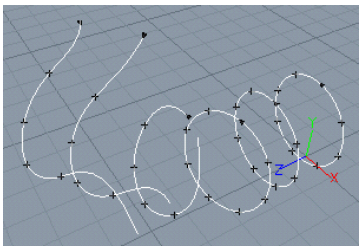


Note

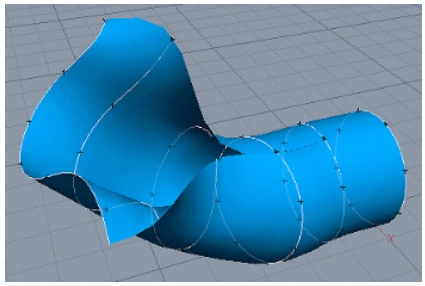
Note: If you need to delete the cross section curves but you don't want to delete the mesh, switch the Constructor of the mesh to None from the property window's Spec tab before deleting the curves.

Mixing open and closed curves

You can freely mix open and closed curves to define a cross section surface. For example, you can create a tube through three open and three closed curves, as demonstrated by the following image:



Three open and three closed curves



The result surface contains both open and closed sections

Extruded Surfaces

The Extrude tool can be used, for example, for creating 3D logos. The tool can handle nested curves by automatically detecting which one of the curves defines holes. You can also control the beveling shape, radius and other extrusion options.

Tutorial project: 'tutorprojects\modeling\nurbs\extrude\logo'

Let's create a simple logo whose depth is 0.1 m and edge rounding is 5 mm.

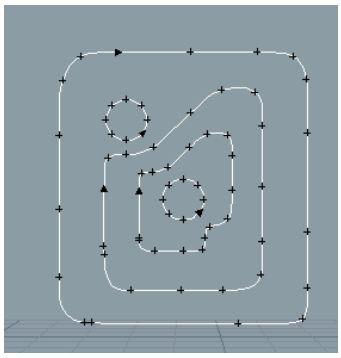
1. Create a set of closed NURBS curves representing a desired logo shape.



Extrude tool

2. Select the curves in any order and click the Extrude tool in the control bar.

3. As usual, the control bar now shows you a number of extrude tool specific options. Set Beveling Type to Rounded and find an appropriate beveling Radius and extrusion Depth. When found, click Accept.



Nested curves for a simple logo



Specifying extrusion options

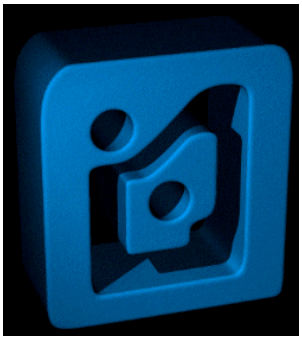
The Extrude tool also creates an Extrusion Manager object. This object controls the beveling radius and other extrusion specific options - they can be changed any time.

For example, to change the type of the beveling of the logo we created above:

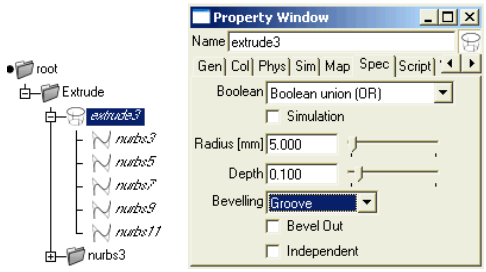
4. Select the extrusion manager object (it is called 'Extrude') and open the property window. Go to the Spec tab and select another beveling type.

The shape of the logo is updated instantly.

You can also modify the original construction curves, which are placed inside the extrusion manager object, to reconstruct the object.



The logo defined by the nested curves



Select the extrusion manager object and change the Beveling type from the property window



Note

The Extrude tool can extrude filled objects only from closed curves. For example, an 'S' shaped curve cannot be extruded that way, because the curve does not have well defined 'interior'. Therefore, the tool cannot determine the direction of the beveling or nesting of multiple curves. Extruding open curves makes sense if the No Lids option is set.



Note

If the Independent option is checked, each selected object is extruded independently into a separate level. By default this option is not set.

If you wish to create holes and you have checked the Independent option, you need to put curves into a level and select the level before extruding. In this case a hole will be created. If just just select your curves and set the Independent option, then each curve will be extruded separately and no hole will be created.

Trimming NURBS Surfaces

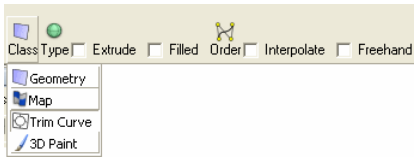
Trim curves can be used for cutting NURBS meshes. For example, one can drill a hole in a NURBS mesh using a circular trim curve. There are several ways to create a trim curve. You can select various trim curve tools from the NURBS tool tab. Also, you can use the Unified tools from the Tools - Unified menu, like shown in the following examples.

The unified creation tools create trim curves with a freeform, circular or rectangular shape.

To create a freeform trim curve:

1. Create a NURBS mesh using the Sweep tool, or simply use the NURBS rectangle tool and set the U and V parameters to a value of 8.

2. Select the mesh and activate the Tools/Unified/Create curve tool from the menu. As usual, you can now see the curve tool specific options in the control bar. Set the Class option to Trim Curve and make sure the Closed option is checked.



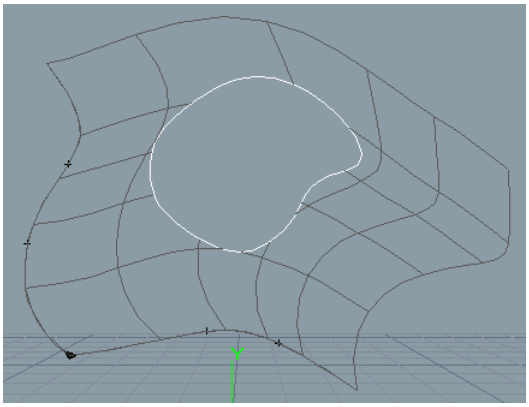
Set the Class option to Trim Curve to create a trim curve

3. Click the mouse over the mesh to define control points for the trim curve. When done, accept the tool.

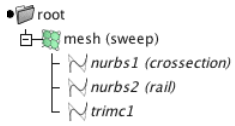


Note

A trim curve is an object whose geometry is defined in the *parameter space* (also called the uv space) of the parent mesh. Therefore, when creating trim curves, the position of the mouse pointer must be within the borders of the parent mesh.



A trim curve created for the selected mesh



A trimmed mesh

The mesh now consists of three curves: the original construction curves plus one trim curve.



Note

If you created the first mesh using the NURBS rectangle tool, the hierarchy consists of only the mesh and the trim curve.

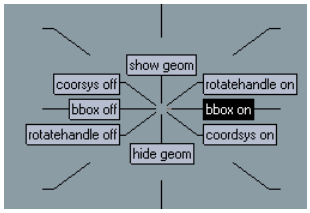
Modifying trim curves

Trim curves can be modified just like ordinary NURBS curves. For example, to move the trim curve to the left top edge of the mesh, just select the trim curve and use the Move tool as usual.



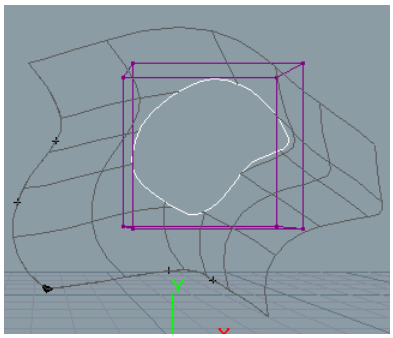
Note

In order to transform a trim curve, you always have to click the mouse **over** the mesh. Otherwise, the transformation tool will not be able to compute the necessary transformation in the parameter space of the mesh.



Switch on the bounding box through 'w' compass menu

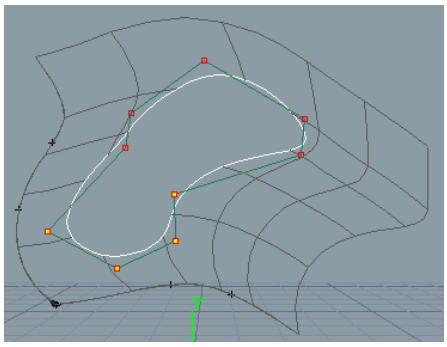
You can also use the standard transformation handles to modify trim curves. As usual, you can switch transformation handles on and off using the 'w' compass menu. For example, you can modify trim curves through their bounding box. To be able to see the bounding box, select the trim curve and right click in the view window, to get the pop-up menu. Select Handles/Bounding box. Then use the 'w' compass menu like shown here. If the planar mesh is flat, the bounding box has a rectangular shape.



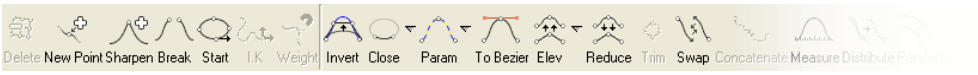
Move the trim curve by dragging the bounding box

Just like regular NURBS curves, also trim curves can be single point edited in the edit mode. As usual, hit the space bar to enter the edit mode and drag points to move them. You can also multi select any number of control points and apply standard transformation tools, such as move, scale and rotate, to the selected points.

When you select a trim curve, the control bar shows you a number of tools which can be used for manipulating the curve. We have already demonstrated most of these tools in the context of NURBS curves.



A trim curve in the Edit mode

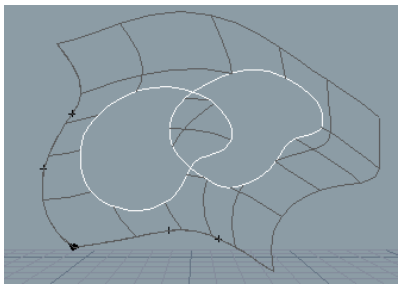


Trim curve specific tools

Trim curves can be duplicated in the same way as any geometric objects. For example, just hold down the copy modifier (Ctrl) while moving the trim curve.

You can copy trim curves using standard Edit/Cut, Paste and Duplicate tools. For example, cut a trim curve from one mesh and paste it inside another mesh. Or, use simple drag&drop in the select window to move trim curves in the object hierarchy. Because trim curves are defined in the parameter space of the mesh, all deformations you apply to the mesh will also deform the trim curve accordingly. For example, if you modify either of the two sweep curves, the trim curve gets modified with the mesh accordingly.

Tutorial project: tutorprojects\modeling\nurbs\trim curves\trimmed mesh



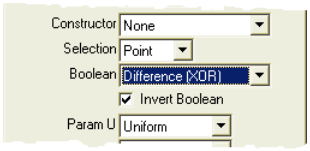
Ctrl+drag to duplicate a trim curve like any other geometric object

Boolean operations of trim curves

The general Boolean operations work also for trim curves. The difference is that since trim curves are defined in the two dimensional uv space', also the Boolean operations are applied in a two dimensional space.

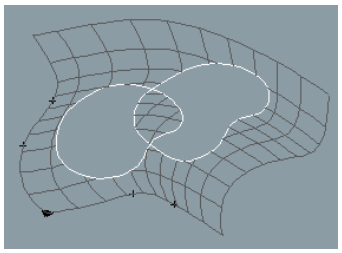
All the three standard Boolean operations are supported: Union (OR), Intersection (AND) and Exclusive Or (XOR).

Boolean operation is an attribute of the actual NURBS mesh. You can select the desired Boolean operation method from the property window's Spec tab.

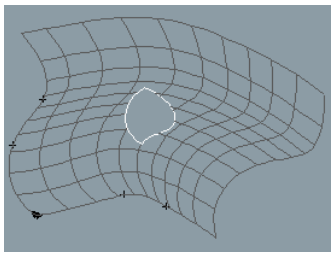


The Boolean attribute of the mesh

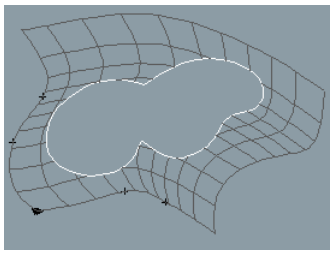
The following three images demonstrate the effect of different Boolean operations.



Difference (XOR)



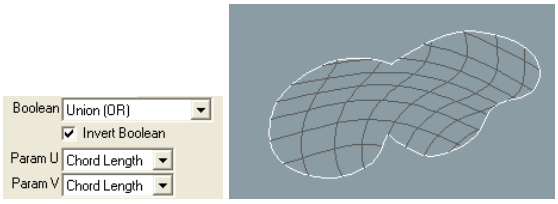
Intersection (AND)



Union (OR)

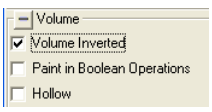
Just like you can invert the Boolean operation of a regular 3D Boolean object, you can invert the Boolean operation between the trim curves of a mesh. Just check the Invert Boolean option through the property gadget of the NURBS mesh.

The Invert Boolean option inverts the inside/outside property of trim curves so that what was inside before becomes outside.



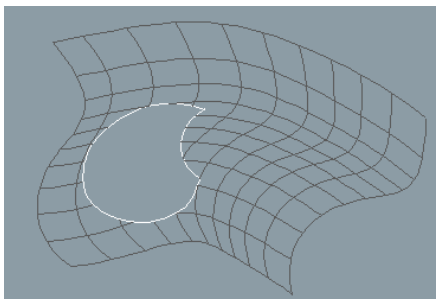
The property window for the NURBS mesh: Boolean operation set to Union and result Inverted

In addition to inverting the top level Boolean operation of trim curves associated with a mesh, you can invert an individual trim curve.



Trimming side inverted

To do this, select the trim curve and go to the property window's Gen tab. Check the Volume Inverted property (see the Boolean operations between true 3D objects).



Trim curve 1 AND NOT trim curve 2

These three attributes give you total control over Boolean operations between trim curves.

Converting trim curves to regular NURBS curves

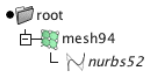
Let's assume you have a trim curve and you need to convert it to a standard NURBS curve. The To 3D tool does this. Just select a trim curve and click the To 3D tool to map the trim curve to 3D space.

This tool creates a regular NURBS curve, which passes through the control points of the trim curve. As a result, the shape of the 3D curve changes! The reason is that the uv space of a mesh is usually curved whereas the 3D model space is neatly linear. The more control points the trim curve has, the more accurately the curves match.

Converting NURBS curves to trim curves

It is also possible to map the control points of a regular NURBS curve to the parameter space of a mesh.

To create a trim curve from a NURBS curve:



Create a NURBS curve inside a mesh

1. Create a NURBS mesh using, for example, the NURBS/Rectangle tool.

2. Create a closed NURBS curve over the mesh. In the select window, drag & drop the curve inside the mesh. You now have a NURBS mesh, which includes one NURBS curve.

3. Select the NURBS curve. The control bar now shows you a number of NURBS curve specific tools. Click the Trim tool.



Note

Use the Trim Parent tool in the second row of the toolbar, not the icon of the Create Trimmed Surface tool described in the next example.



Note

In this operation, the shape of the created trim curve does not usually match the shape of the original curve exactly. The more control points the curve has, the more accurately the trim curve follows the original NURBS curve.

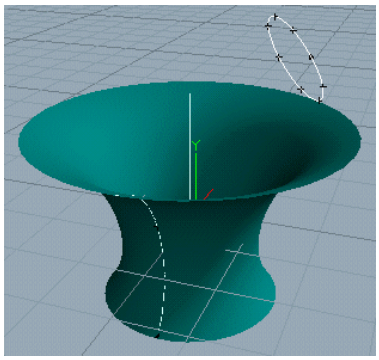
Projecting curves to trim curves

It is also possible to convert a NURBS curve to a trim curve by using the current transformation of a view window. In this method, the control points of a NURBS curve are not snapped to the nearest surface points. Instead, they are projected to the surface behind them.

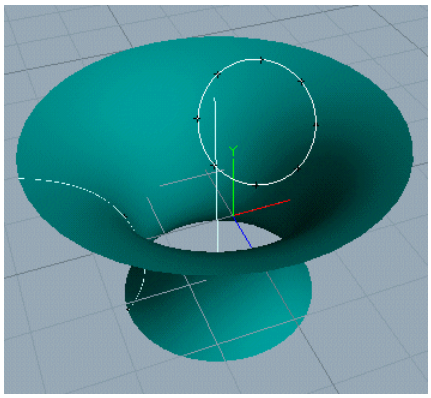
1. Create a curved NURBS mesh and a closed NURBS curve, as shown in the next picture.

2. Select the mesh and the curve. Then rotate the view window to find a desired projection.

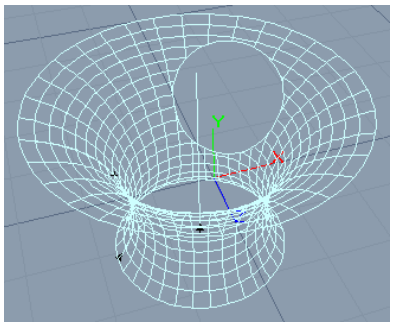
3. Then select the Tools/Nurbs Construction/Create Trimmed Surface tool and click Accept.



A NURBS surface and a NURBS curve



Rotate the view window until the curve is in right place over the mesh



A NURBS curve projected onto a mesh

Welding

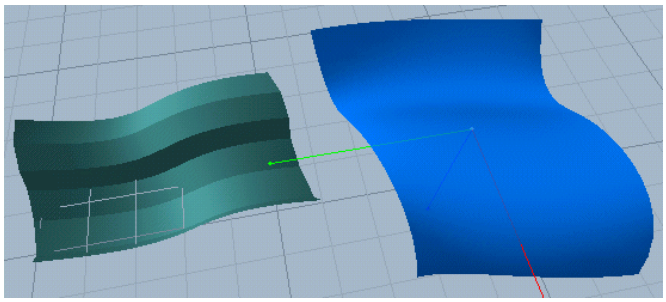
The Welding tool connects two surface edges with a special purpose weld object. Basically, the weld object is a NURBS mesh with some additional intelligence so that it maintains a seamless joint between the specified edges.

Any two edges with different point counts, orders and other properties can be welded.

To connect two meshes, one polygonal and one smooth surface, by welding:

1. Create two NURBS meshes: a polygonal one and a cubic one.

2. Select both meshes. The control bar now shows you the weld tool.



Two NURBS meshes with a different order and density

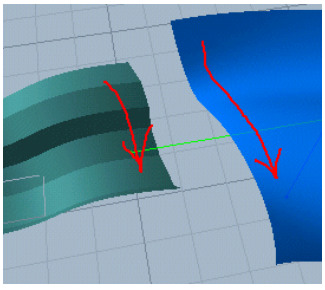
 *Weld tool*

Activate the tool. The tool automatically welds the two surfaces. You can change the weld by dragging a line over the mesh, near the edge you want to weld. The line you draw also defines the direction of the weld mesh.



Note

You have to drag **over** the meshes (inside the mesh boundaries), as shown in the example image



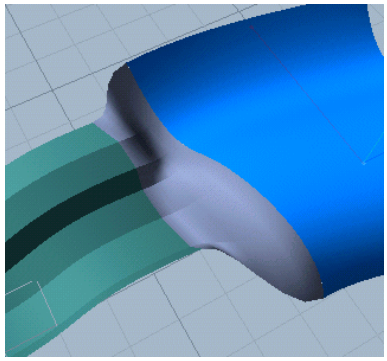
Mark two edges to be welded

If you accidentally marked a wrong edge, you can drag again to fix the error. There is no need to restart the tool.

3. When the desired edges are welded, click Accept to finish the tool. The two meshes are now connected.

You can modify the original meshes after welding. The shape of the welding surface is updated accordingly.

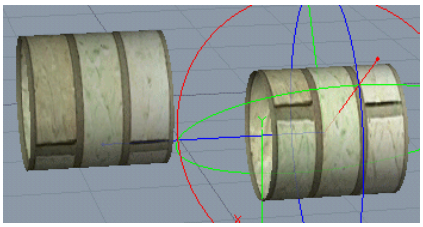
Tutorial project: 'tutorprojects\modeling\nurbs\weld\weld'



A polygonal and a cubic mesh welded

When welding closed surfaces (such as pipes, for example), there is an additional freedom to control the twist of the welding surfaces (the positions where the 'u=0' isoparametric curve meets the surfaces).

Let's connect two NURBS cylinders with each other to demonstrate this.

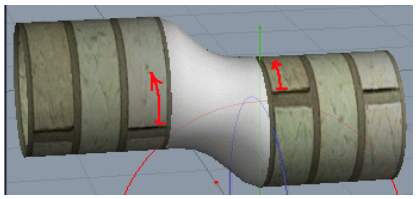


Create two NURBS tubes

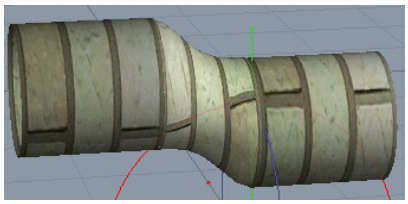
1. Use the NURBS/Cylinder tool to create two NURBS tubes (reset the Filled option).

In order to see the twist effect, switch to the wire frame rendering mode.

2. Select both NURBS tubes. The control bar now automatically shows you the Weld tool. Activate the tool and drag along the edges to be welded, then Accept the tool.



Mark an edge from both cylinders. Drag points determine the start and end points for the first isoparametric curve of the welding mesh.



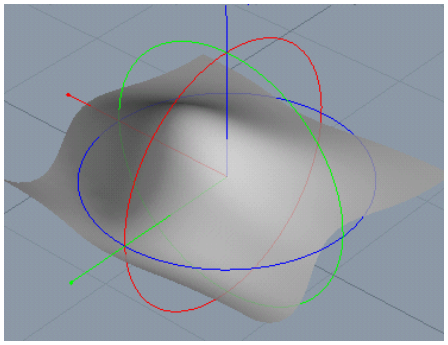
Welded tubes

Welding surfaces and trim curves

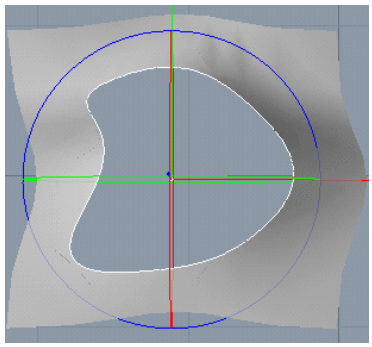
Welding tool also allows you to weld the edges of trimmed surfaces.

1. Create a NURBS surface.

2. Select the surface and select Unified/Create Curve from the Tools drop down menu. Set the Class option to Trim curve and enter a desired number of control points over the NURBS surface to trim it.



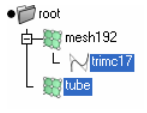
A NURBS surface



A trimmed surface

3. Use sweep or another appropriate tool to create a NURBS tube.

4. Select the trim curve and the tube and activate the Weld tool from the pull-down menu Tools/NURBS Construction/Weld Edges.



Select the trim curve and the tube mesh



Note

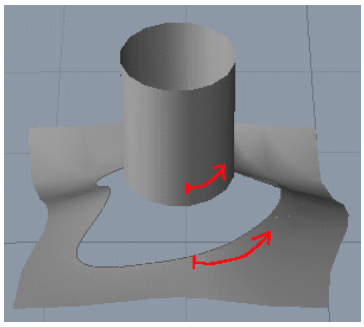
A mesh can contain any number of trim curves. You have to select the actual trim curve to which the weld mesh should be attached!

Mark an edge from both surfaces to weld them.

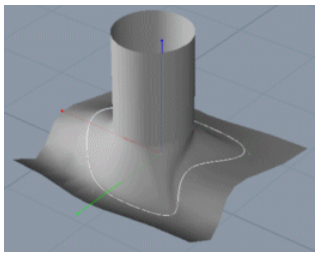


Note

Both edges are closed - by controlling the drag start points, you can control how twisted the weld surface will be.

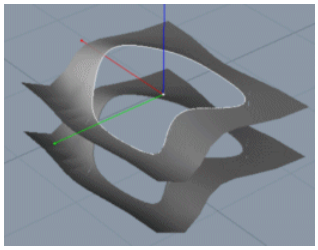


Mark the edges to be welded



The trimmed surface edge welded with the tube

Welding two trimmed surfaces

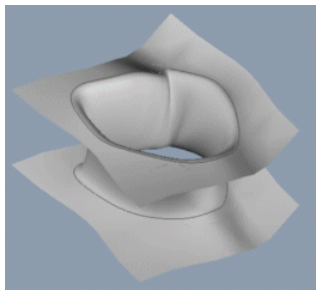


Two trimmed meshes

The Weld tool can also weld two trimmed meshes with each other.

1. Create two trimmed NURBS surfaces somewhat parallel to each other.

2. Select both trim curves, click the Weld tool of the NURBS tool tab and mark edges to create a weld surface.

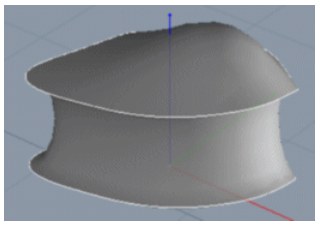


Two trim curves welded

The trimmed edges are now smoothly joined with each other. Let's play with this example a little bit more.

3. Select both trim curves and click the Invert tool in the control bar (it's located at the far right end).

This inverts the inside/outside property of the selected trim curves and they trim out the outer part of the surface.

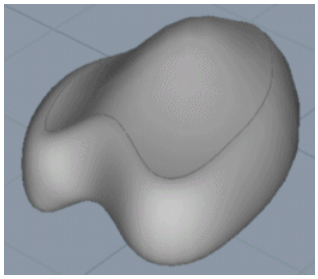


Trim curves inverted

4. Select the weld surface and open the property window. Go to the Spec tab. Here you can see two Size fields, which define how sharp the weld surface is at its boundaries. The smaller the value, the sharper the joint.



The radius fields of the weld surface



Weld surface with negative size attributes

If you want completely sharp edges, set these values to zero. We just want to change the direction of the weld surface (the edges of the weld surface should point outwards rather than inside).

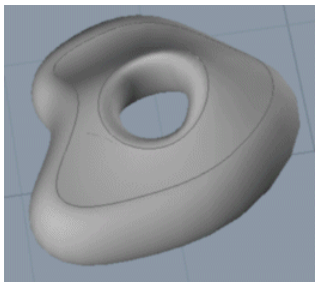
Change the sign of both the values (make them negative).

Tutorial project: 'tutorprojects\modeling\nurbs\weld\welded trims'



Note

You can create any number of trim curves to both surfaces and connect them with the weld tool. Also the weld surface itself can be trimmed and welded with another weld surface.



Two trimmed edges connected with the weld tool

Controlling the accuracy

When connecting the edges of the surfaces, the weld surface is capable of following the shape of both the edges exactly. However, when welding trim curves, it is mathematically impossible to create an exact weld surface for the trim curve.

If the joint is not accurate enough, increase the number of control points of the trim curves. The higher the point count of the trim curves, the more accurate the weld becomes.

Fillets

Let's imagine you have two NURBS surfaces representing a body and a handle of a coffee cup and you would like to blend the handle to the body seamlessly. The fillet tool can be used for this.

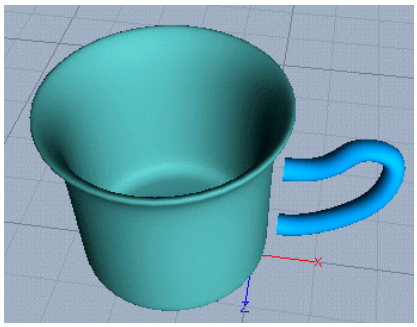
Let's try this.

1. Use the rotation tool to create a body of a coffee cup. Use the Sweep tool to construct the handle. Position the handle to its correct position, as shown by the example image.

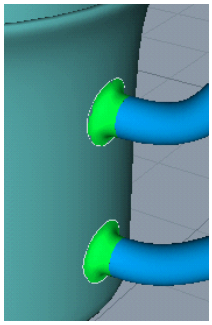


Note

Make sure both ends of the handle point towards the body of the cup.



Two meshes for blending



The handle blended with the body

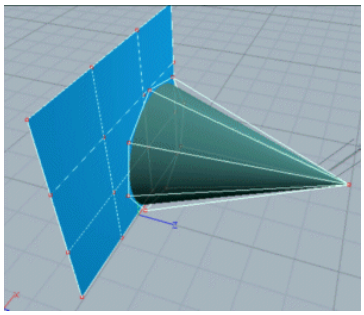
2. Select the body and the handle (in this order) and click the Fillet tool. You can now see the fillet surfaces created between the body and the handle. As usual, the control bar shows you fillet specific options, such as the radius. Specify the desired radius and click Accept.

You can now move the handle (or the cup) and the shape of the created blending surfaces is updated. The handle will remain seamlessly blended with the cup.

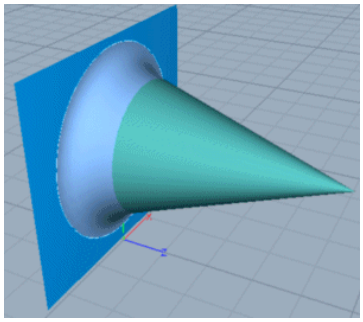
Tutorial project: 'tutorprojects\modeling\nurbs\fillet\cup_handle_fillet'

The fillet tool can determine the shape for the blending surface by extrapolating the second selected object (the handle). The following example demonstrates this.

1. Create a 4x4 rectangular mesh. Then use the rotate tool to create a cone. Place the open end of the cone near the planar mesh as shown in the image:



A cone and a planar mesh



2. Select the mesh and the cone and activate the Fillet tool. Click Accept.

The created surface is tangential with both the cone and the planar mesh.

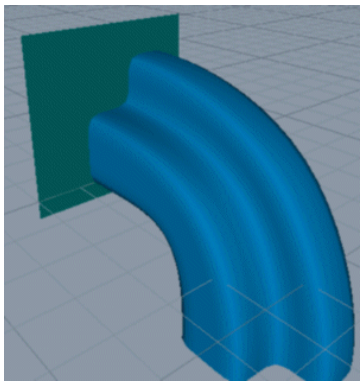


Note

If the cone is too big in relation to the planar mesh, the fillet tool may not be able to connect it to the planar mesh.

Tutorial project: 'tutorprojects\modeling\nurbs\cone_mesh_fillet'

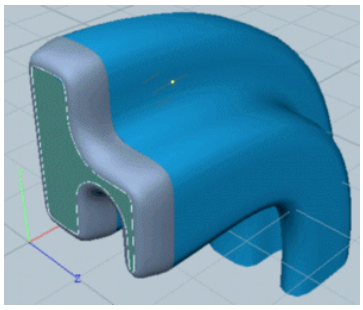
The fillet tool can also be used for creating a lid for the ends of a closed profile tube surface.



A planar surface and a tube

1. Create a closed profile curve and use the sweep tool to create a tube. Then create a planar surface, which will be used as a lid for the tube.

2. Select the planar mesh and the tube and activate the Fillet tool. Set the Trim option to Trim curves and this time specify a negative Radius. This makes the created surface bend inside rather than outside. Accept the tool.



A sweep surface blended to a planar mesh with a negative radius

3. Select the trim curve created inside the planar mesh object. The control bar now shows you the trim curve specific tools. The Invert option is set for the trimcurve. Uncheck the Invert option to change the trimming side.

Tutorial project: 'tutorprojects\modeling\nurbs\fillet\tube with lid'

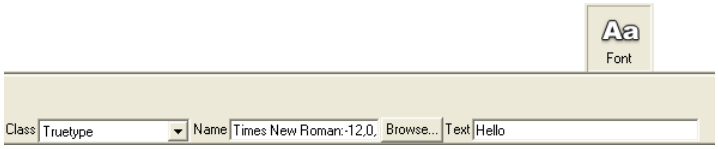
The Font Tool

The *Font* tool creates font outlines. Outlines are represented as regular NURBS curves. You can use them with all surface construction tools.

Let's experiment with the tool.

1. Activate the Font tool. It's in the NURBS tab.

You can now see a number of tool specific options in the control bar.



The Font tool activated

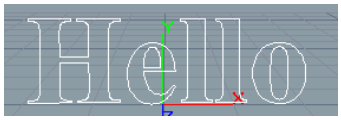
2. Specify the desired font properties and type in a string 'Hello' into the Text field.



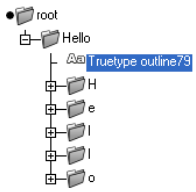
Note

Use the Browse button to bring up the standard font selector window and select a font, style and size.

3. Then enter two points through the view window to define a base line for the outline curves. When the second point is entered, the tool is automatically accepted and the curves are inserted to the scene.



Hello using Times New Roman

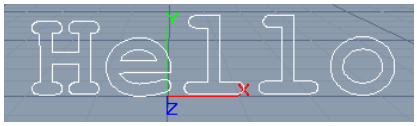


Select font manager

4. Let's assume we made a mistake. Instead of using the Times font, we should have used the Courier font.

In the select window, you can see an object whose name matches the text you specified into the Text field ('Hello', in this case). Open this level and you can see a font manager object. The font manager object is named according to the font class you specified (Truetype, for example). Select the font manager object.

5. Open the property window. In the Spec tab, change the font type to Courier New, for example. The curves in the view window are instantly updated to reflect the new selected font.



Hello with Courier font

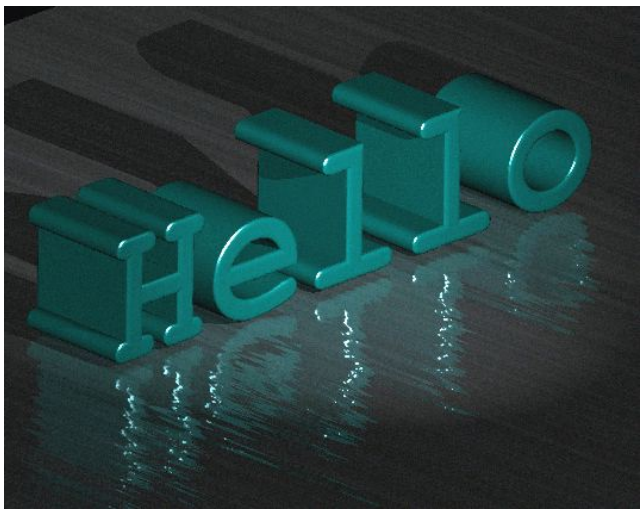
Let's end this tutorial by creating a 3D logo from the curves. The Extrude tool can be used for this.



Extrude tool

6. Select the entire 'Hello' folder from the Select Window. Click the Extrude icon in the NURBS toolbar.

7. Select and specify the desired extrusion options through the control bar. For example, set Beveling Type to Round. When done, click Accept to finish the tool.



Extruded outline fonts on a semi reflective floor

8. Drag and drop a material such as 'Gold' from the Select Window's material tab into the view window to assign a nice material to the new 3D text object. Add a point light source above the text. Use the Analytic Rectangle tool to model a floor below the text. Hit '0' key to render the view and check the result.

Tutorial project: 'tutorprojects\modeling\nurbs\fonts\hello'

Vectorizing Curves from Images

The Vectorize tool reads an image as an input, traces out contours from it and generates a set of NURBS curves which represent the traced contours.



The Vectorize tool

The generated NURBS curves can be used in various ways. For example, you can render them directly using the photo realistic rendering engine. Or, you can extrude the curves to get a 3D logo.

Let's imagine you have a photograph of a logo. You have scanned the photo to an image and now you want to create a 3D logo from it. Do this:

1. Activate the Vectorize tool. The control bar now shows you a set of tool specific options.

2. Use the Browse button in the control bar to locate the image to be vectorized.

Because the selected image is not an outline image, set also the Edge filtering option. If you have an image which already contains, say, black outlines on a white background, you do not need the Edge filtering option.

If the Fit to NURBS option is set, the tool runs a simplification procedure. It tries to represent the contour curves with a minimum number of control points. If this option is not set, the tool creates a point for each pixel.



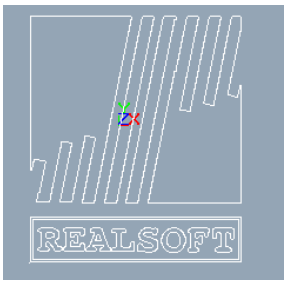
Specify options for the Vectorize tool

3. Click Accept. The tool creates a new level object, which contains the generated curves.

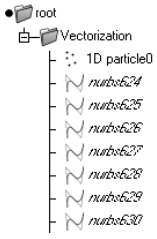


Note

Images can contain bright pixels which are not connected to any other pixels i.e. they form a point rather than a curve. The Vectorize tool uses 1D particles to represent these individual pixels.



The image vectorized



The generated curves can then be rendered by resetting their Invisible in Photo Realistic Rendering option. Or, the curves can be extruded by using the Extrude tool.



Curves rendered with the Particlefur material

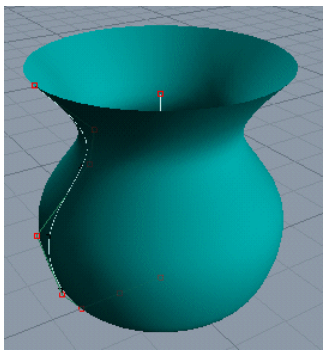


Extrude tool applied

Creating Curves from Surfaces

The Curves from Surfaces tool can be used for creating a set of NURBS curves from a surface with a specified accuracy. The curves can then be used for various purposes.

In this tutorial we will create a vase which consists of numerous fibers. The fibers are texture mapped NURBS curves.



Create a NURBS vase

1. Use the Rotate mesh construction tool to create a vase.

 *Curve extraction tool*

2. Select the vase and apply the Curve extraction tool from the NURBS tool tab. Use the control bar to specify suitable options.

Use the U Curves and V Curves options to specify whether curves should be evaluated in the U, V, or both the directions of the mesh.

The Curves field specifies the curve extraction density.

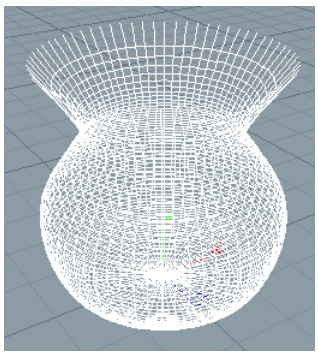


Note

This option doesn't define the total number of curves but the density. The value 1 gives the minimum number of curves. The higher the value, the higher the curve count.

The Points field specifies the number of control points per curve. Again, this is not the absolute point count but specifies the density for point insertion. The higher the value, the more points will be created to the curves.

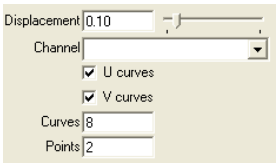
Check both U Curves and V Curves options. Set Curves to 8 and Points to 1. Then Accept the tool.



The vase converted to a set of NURBS curves

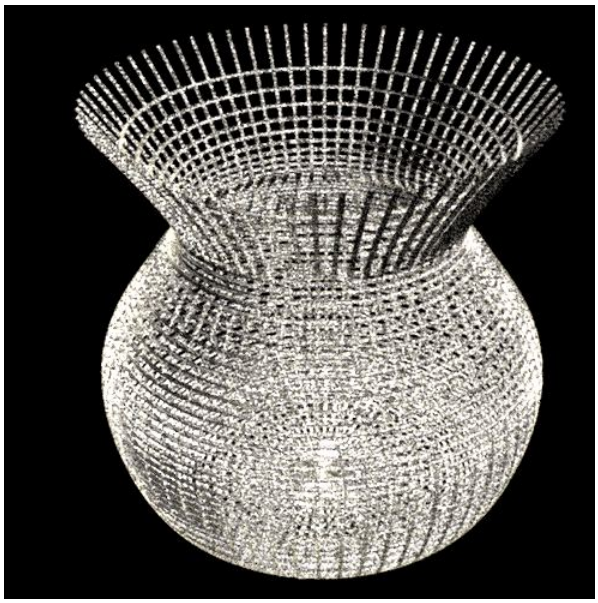
3. Select the root object and drag & drop the 'Particlefur' material into the view window. You can also use another material to define a desired color and other properties for the vase.

4. Select all curves (select the first curve, scroll down, press and hold the Ctrl key and select the last one), open the property window and go to the Gen tab. Uncheck the Invisible in photorealistic rendering option. Render the view.



Property window of the curve manager object

If you need to modify the number of curves, select the Curve Manager level object from the select window and use the property window to specify a new curve count.

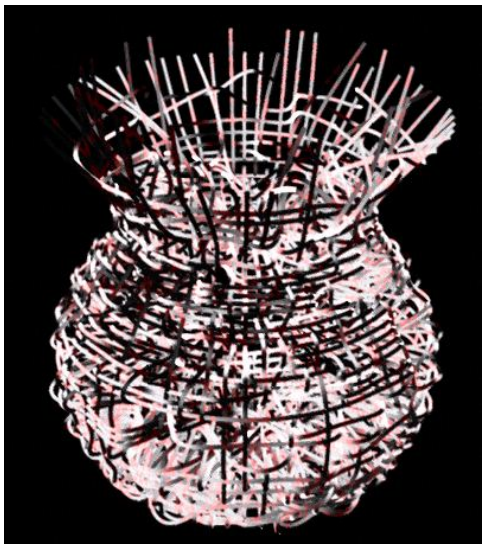


A basket created from texture mapped NURBS curves

The Displacement attribute displaces the curves from the surface.

The Channel selector allows you to use any material to modify displacement values. To use this feature, map a material to the curve manager and specify a channel, such as color, which controls the density. The given channel is evaluated over the surface and values are used to scale the displacement defined by the Displacement option.

Tutorial project:
'tutorprojects\modeling\nurbs\surfacetocurves\vase'



Non-uniform displacement defined by the Spotty material

The Displace Tool

The Displace tool uses a bitmap image to define a translation for the control points of the selected NUBS meshes. You can use the tool to create real bumps to mesh surfaces (the material bump mapping modifies only the shading of the surfaces, not the geometry).

The following example shows how to use the tool.

1. Go to the NURBS tool tab. Activate the Rectangle tool.

2. Define a suitably high resolution for the mesh in the tool control bar. The displacement tool displaces the existing points of a mesh, so only a dense mesh can represent bumps defined by typical image resolutions. We will use a low-resolution image, so U=50 and V=50 should suffice for our test.

3. Finish the Rectangle tool by drawing a square mesh in a view window.

4. Make sure the mesh is selected. Go to the NURBS tool tab and activate the Displace tool.

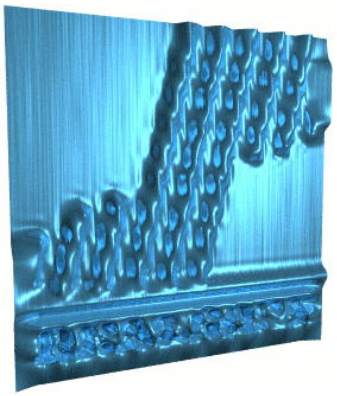


The Displace tool activated

5. Define a suitable maximum height for displacement using the Displacement field in the tool control bar, for example 0.1 meters.

6. Browse a suitable image file to the Image field, for example 'textures/logo_bw'. Now the program computes the displacement and you can see the result in the view window. Adjust the height if necessary. Then click Accept to finish the tool.

The result is shown here as a ray traced image. The 'brushsteel' material was slightly modified and mapped to the displaced surface to make it appear more interesting.



The displaced surface rendered as brushed steel



Chapter 6. Modeling Buildings

Creating Parametric Objects

The Packer object and its derivative Replicator provide tools for parametric object creation. Parametric objects can be rescaled and they keep the desired component dimensions. A kitchen cabinet is a good example of an object, which is frequently needed in various sizes. Rescaling a cabinet must not change the thickness of the construction material (wooden boards).

Modeling a parametric object is not a simple or automatic process. The shape must be divided into horizontal, vertical and depth layers. Behavior of each component must be defined. Inclusion of material mapping objects is wisest to plan in advance. These additional efforts in the modeling phase are rewarded later, because the reusability of the object is much greater than of models, whose dimensions are fixed.

As an example, we will create a complete window object shown in the attached image.

Tutorial level: Medium/ Advanced

Example project: tutorprojects/modeling/ packer/step1-step9

A window can be divided into depth layers in a natural way:

- Outer frame (mouldings outside the house)
- Panes and the surrounding panels
- Interior mouldings

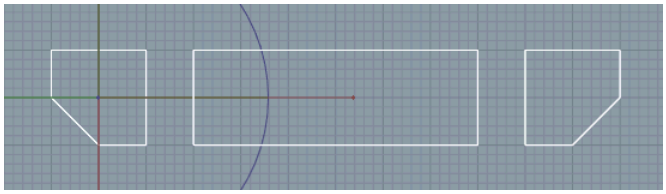


Outer Frame

Take a front view and enable parallel projection. Drag and drop a suitable grid, say 1 cm, from the grids tab of the select window into the view window. Zoom the view so that you see approximately one meter of space. We will model the window in its minimal size.

The topmost board of the outer frame of our example window has beveled edges. The beveling angle, 45 degrees, must not change when the window is resized. Therefore, we must split the board into 3 sections.

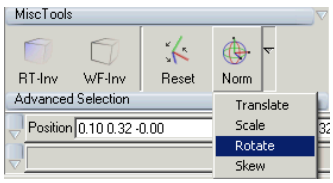
Activate the Analytic/Polyhedron tool from the tool bar. Draw the left end of the board as shown in the picture on the right. After the fifth point, click Accept to finish the tool. Change the depth of the board to 0.02 meters using the tool bar control.



Activate the Analytic/Cube tool and draw the middle section. Make it 30 cm wide and 10 cm high. When the cube is drawn, change its depth to 0.02 meters. Then duplicate the polyhedron at the left end and use the Transformation/Mirror tool to flip it horizontally. The gaps between the parts can be 0.05 meters at this stage. Do not make them much wider because they will contribute in the minimal width of the frame.

Let's rename the components so that it will be easier to manage them. Slowly double click the name of the first polyhedron in the select window and rename it as 'left'. Rename the cube as 'mid' and the second polyhedron as 'right'.

Multi select all three objects (selection order is not important). To make sure that geometry will be arranged in the correct direction, open the MiscTools bar below the select window, click the little triangle at the right side of the Norm tool and select Rotate from the list. The coordinate handles of all three objects become aligned with world's directions. Especially polyhedrons may have unusual handle directions because they include some diagonal lines. The packer tool, which we will apply next, will take its packing directions from the first selected object.



Make sure packed objects have proper handle directions

Having all three objects selected, go to the Measure tool tab and activate the Packer tool. The default options are suitable; Orientation=Detect means that the tool will examine how the selected objects are initially arranged

and pack them accordingly. In our case, the objects are placed side by side along the X axis, and the tool will select X directed packing accordingly. Hit the Accept button to finish the tool.



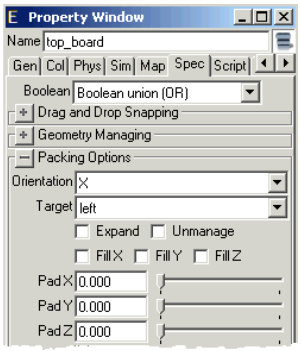
Note

Packing order is defined by the pivot points of selected objects. Pivot point is the center point of the transformation handles (3 RGB colored circles and axis lines), which become visible when the object is selected. The pivot point can be moved by dragging it with the mouse. You can guide the packer tool by arranging the pivot points in a suitable way.

A new item packer appears to the select window. The selected parts were placed to its sub hierarchy. In the view window, you will see a box-like container, which includes the components. Little arrows labeled with X and Y show the orientation of packing. Objects are packed along X direction as planned. Select the packer and rename it as 'top_board'.

Make sure the packer object is still selected and open the property window. Go to the Spec tab and open the Packing Options frame. The Target selector lists the three components managed by the packer. Select Target=left. Then clear the Expand option, if it is set. The packer will not expand the space reserved for the leftmost part proportionally to fill the packer space. The left part will occupy only the minimal space. Consequently, the left part is placed to the left end of the packer.

Pick the 'right' object from the Target list of the packer gadget and clear the Expand option if set. Then select 'mid' from the Target list. Enable Expand, as the middle part should indeed expand to fill the gap between the left and right ends. Furthermore, activate the Fill X Option, so that not only the container space of the middle part but also the actual geometry expands.



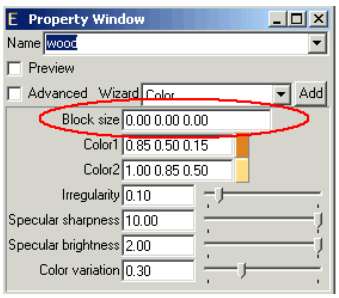
Packing options for the leftmost part

The top board packing is now ready. The three parts should now be joined without gaps. The tool bar shows the width of the packer. Try changing it from the current value (should be about 0.6 meters) to say 1 meter. The middle part stretches to fill the increased dimension. Then set the width to 0.6 meters.

We have so far created a very simple object, but it already shows the essence of the packer object: ability to easily control stretching and placement of components in rescaling. The top board needs a couple of finishing touches before it is ready for use. First of all, let's hide the packer box to make the wireframe simpler and cleaner. In the property window, go to the Wire tab and activate the Hide Geometry option.

Next we will assign a material to the board. Unselect all objects by clicking empty area on select window. Take a side view. Switch to the material tab of the select window. Select wood from the library, and then activate Map/Parallel from the popup menu. Press the Shift key down and draw a symmetric cube in the view window. Shift key helps to keep all edges of the map cube equally long, which is important for the procedural wood material. Draw the cube about 0.05 meters wide and high. The second click finishes the parallel map tool.

Take a front view and hit the render button to make sure that the board is made of pine wood. You will probably see some unexpected seams in the wood pattern. This is because the procedural wood is configured for parquet floors and other similar wide surfaces made of several parts. To fix the problem, use the property window and change the Block size of the wood material to ' 0 0 0 '. The board will become like cut out of from a single trunk.





Continuous wood

Hint: Rotating the parallel map slightly from the top view may improve the look of the pine wood pattern, because the board is no longer cut perfectly along the wood grains. Real planks are cut that way because of the conical tapering of trunks.

Switch back to the leftmost Geometry tab of the select window. Drag the parallel mapping object into the hierarchy of the top_board packer. Select the parallel map and apply WF-inv tool, which is in the MiscTools bar below the select window. This hides the wireframe of the material map object. We do not want it to clutter the clean wireframe of the finished object. The board element is now ready.



The hierarchy of the finished top board

We will need three more boards for the lower part of the frame. This time we will not make any corner cuts or other decorations, so the modeling process is a just a simpler version of the steps above. Take the front view, activate the Analytic/Cube tool and draw a cube, which is 0.1 meters wide and 0.4 meters high. Remember to set its depth to 0.02 meters.

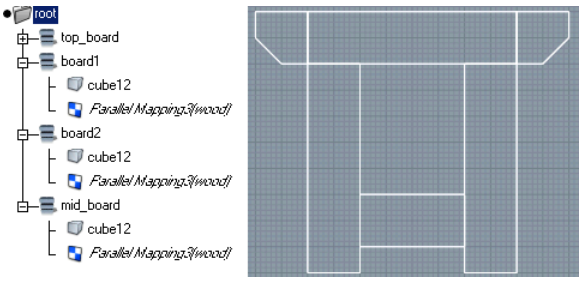
The new cube selected, activate the packer tool. Click Accept to finish the tool with the default options. Rename the new packer as 'board1'. Open the property window and enable Packing Options/Fill Y and Expand for the new cube. Set Wire tab's Hide Geometry option for the new packer.

Go to the select window and select the parallel map object under the top_board packer. Press Ctrl key down and drag the parallel map into the board1 packer. Ctrl dragging creates a copy of the object. Find and select the created copy from the hierarchy of the board1 packer. Apply MiscTools/WF-Inv button to make it visible again. Use the circular transformation handles to rotate the parallel map 90 degrees, so that wood grains match the direction of the board. Move the mapping on top of the new board. Make the map WF-invisible again.

Duplicate the board1 by Ctrl-dragging its red translation handle in the view window. Place the copy to the opposite side of the frame and rename it as 'board2'.

Hit Ctrl-d hotkey to duplicate board2. Press shift key down and rotate the new board 90 degrees using the circular rotate handle (Shift locks rotation to 45 degree steps). Rename the new horizontal board as mid_board and place it between the vertical boards.

The horizontal board is most likely too long to match the gap. Try entering the correct measure to the Height control in the tool bar (Height controls the length of the rotated board). The board does not become shorter. This is because the packer defines a minimal size, to prevent accidental overlapping of packed parts. To redefine the size, open the Geometry Managing frame in the property window and set the Min. Height value of mid_board to 0.2 meters. Then select the child cube of the mid_board packer and change its Height to 0.2 meters. Now the middle part matches the gap.

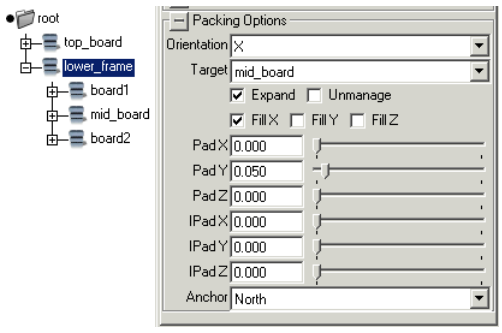


Three new boards added

Select board1, which has a proper orientation, then multi select board2 and mid_board, too. Activate the Packer tool and Accept it. Note how the new packer moves the middle board upward, because the default alignment option is Center.

Rename the new packer as lower_frame. In the property window's Spec tab, set Target=board1. The leftmost board should not expand in the packing direction. Therefore, make sure that the Expand option is not set. Set Fill Y option, because we want the board to fill the packer vertically. Then set Target=board2 and apply the same options: Expand off, Fill Y enabled.

Select mid_board from the Target gadget. The middle part will be the expanding component, so enable Expand. This part is supposed to stretch horizontally, so enable the Fill X option. Select Anchor=North from the bottom of the property window. The mid_board moves to the bottom of the packer. To add a suitable offset, set Pad Y = 0.05 meters.



Structure of the lower frame

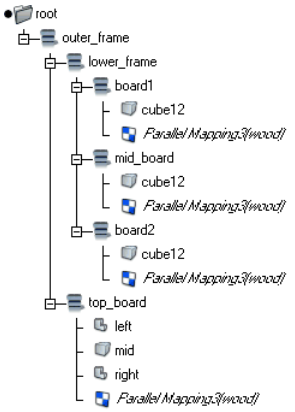
Try changing width and height of the lower_frame packer from 0.4 meters to say 0.6 meters, just to test that rescaling works as expected. Do not worry if lower frame expand over the top board; the next packer level will fix that. If scaling does not work as expected, recheck all packing options of sub objects and make sure they match the instructions above.

Restore the dimensions back to the original size 0.4 * 0.4 meters. Use the wire tab of the property window to hide the wire frame box of the packer.

The next step is to combine the lower frame and the board on the top. Make sure that the pivot points of these two objects are properly centered. In other words, pivots should be approximately on the same vertical line, so that the packer tool will understand that the parts should be packed vertically. Drag the pivots to the center points if necessary, or use the Center Pivot tool.

Select both objects, activate the Packer tool and click Accept. Rename the new packer as 'outer_frame'. Set the packing options of its child objects as follows:

- Set Target=top_board, clear Expand and enable Fill X
- Set Target=lower_frame, enable Expand, set Pad X=0.1 and then enable Fill X and Fill Y
- As usual, hide the geometry of the new packer



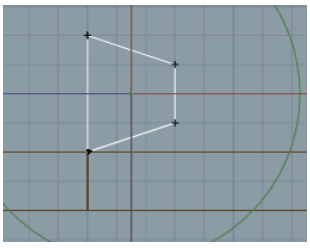


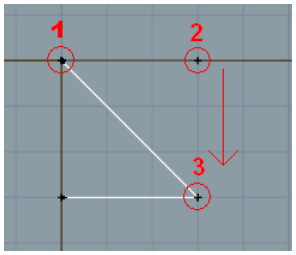
Outer frame of the window finished

The outer frame is now ready. Change the width and height dimensions to w=0.8 m, h=0.65 m. Check that the frame scales properly. Save the project to backup your work.

Window Panes

The frames of the window panes can be modeled easiest using NURBS modeling tools. Take a top view and zoom in to see the left edge of the exterior frame better. Activate the NURBS/Curve tool. On the second row of the toolbar, set Order=Polygonal and enable the Closed option. Draw the profile of the window frame for example as shown in the picture below. After the 4th point, click Accept. You can naturally define a more sophisticated profile, if you like. Our example profile is 3 cm wide and 4 cm deep.

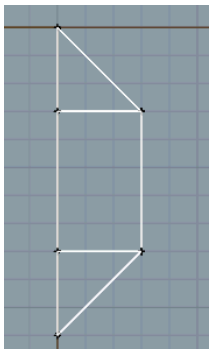




The upper profile curve is skewed 45 degrees

Take a front view. Move the created profile curve to the window corner. We will join frames using a diagonal beveling. Packer-managed assembly needs a corner piece, which is modeled as follows. Duplicate the created profile 2 times (Ctrl-d hotkey). Move the first copy 3 cm (=width of the profile) down from the window edge. Select the second copy, activate the Transformation/Skew tool, click on the left end of the profile and then on the right end. Move the mouse down until the profile is skewed 45 degrees. Click third time to finish the tool.

Select the two duplicates of the profile curve. The right end of the second toolbar row shows X-Sect tool. Activate it and click Accept. A new NURBS mesh appears to the select window. Rename it as 'corner'.



Three NURBS meshes

Duplicate the original profile again. Move the original profile below the corner piece, and the new copy about 5 cm below the original profile. Select both curves, and use the X-Sect tool to create a new surface. Rename it as 'mid_part'.

Duplicate the corner object. Use the tool Transformation/Mirror to flip the copy vertically. Place the mirrored copy exactly below the other two parts, as in the illustration.

Select all three NURBS meshes. The beveled corner pieces may have strange object space orientation. Go to the MiscTools bar, open the side menu of the Norm tool and select Rotate to fix this. Then activate the packer tool and click Accept. Rename the new packer as 'vertical_unit'. Set the Hide Geometry option in the property window's wire tab.

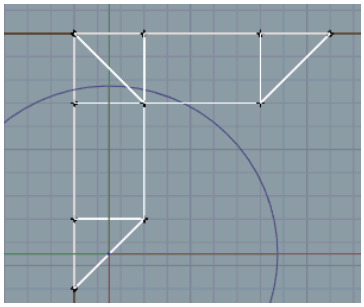
Select the new vertical_unit packer, open the property window and go to the Spec tab's Packing Options frame. The child objects should be packed as follows:

- Set Target=corner and clear the Expand option.
- Set Target=copy of corner and clear the Expand option
- Set Target=mid_part and enable the Fill Y option. Expand should also be enabled.

As usual, test the packer by increasing the height. Only the middle unit should stretch. Reset height back to the original value (0.11 m).

Deselect all objects, take a top view and switch to the material tab. Select wood material and start the Map/Parallel tool from the popup menu. Draw a 5 cm * 5cm box around the vertical_unit. Remember to hold Shift key down so that parallel map becomes symmetric in all dimensions. When the parallel map is drawn, switch to the geometry tab of the select window and drag and drop the parallel map object into the hierarchy of vertical_unit. Finally make the parallel map WF-Invisible from the MicsTools bar.

Go back to the front view. Duplicate vertical_unit, rotate the copy 90 degrees and move it so that the units form one half of the frame, as shown in the picture below. Rename the copy as 'horizontal_unit'.



Joining diagonally beveled pieces using a packer is a bit problematic. The control boxes of the two packers actually overlap in their current position. Normally, packer places the child objects side by side. We can pack the overlapping parts together permanently as follows.

Select the vertical_unit packer. Start the Packer tool and Accept. Rename the new packer as vertical1. Open the property window and select vertical_unit as Target and set the Packing Options as follows:

- Expand enabled
- Set Pad Y = -0.03 meters (=width of the profile)
- Enable the Fill Y option

Thanks to the negative Y padding, the corner pieces stick out from the control box of the packer. Hide the geometry of the new packer in the Wire tab of the property window. The height of the vertical1 increased when Y padding was changed, so move the pieces to match them seamlessly again. Then duplicate the vertical1 packer, rotate or mirror the copy 180 degrees and place it to the other side of the pane. Rename it as vertical2. Duplicate the horizontal_unit as well, rotate the copy 180 degrees and move it to the bottom to complete the frame. Rename the last item as horizontal_unit2.

Activate Analytic/Cube tool and draw a cube, which fills the hole inside the frame. Use the Depth control of the second row of the tool bar to make the cube thinner, for example to 0.005 meters (=5 mm). While the cube is selected, go to the material tab of the select window, drag the material glass and drop it into the view window. This assigns the glass to the cube. Switch back to the geometry tab of the select window. Take a side view and move the cube in the middle of the frame. Rename the level, which contains the cube and the glass material map as 'glass'.

Select both vertical pieces and the glass level. Apply MiscTools/Norm/Rotate to make sure that object spaces are properly aligned. Then start the Packer tool and click Accept. Rename the new packer as mid_frame. Hide its control box geometry in the Wire tab of the property window. The packing options of the children are:

- vertical1: Expand disabled, Fill Y enabled
- glass: Expand, Fill X and Fill Y enabled
- vertical2: Expand disabled, Fill Y enabled

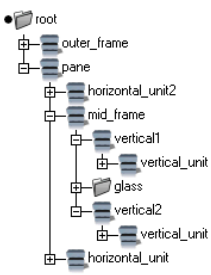


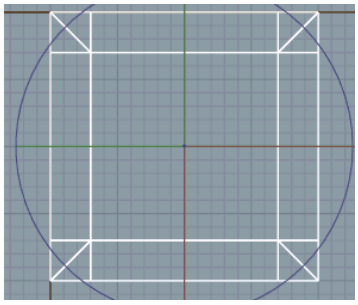
Note

If the frame profile includes a groove for the glass, wrap the glass object into a packer and apply negative X and Y padding to it. Then the glass extends out from the packer into the groove of the surrounding frame.

Finally, select horizontal_unit, mid_frame and horizontal_unit2. Apply MiscTools/Norm/Rotate. Create a new packer with the default options. The packing options are:

- horizontal_unit: Expand disabled, Fill X enabled
- mid_frame: Expand, Fill X and Fill Y enabled
- horizontal_unit2: Expand disabled, Fill X enabled





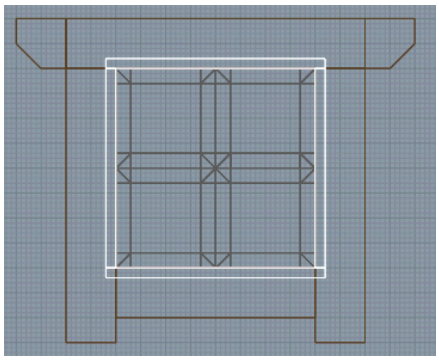
One window pane is now ready. Rename it as 'pane' and adjust its Width and Height to fill one quarter of the whole window (0.2 * 0.2 meters in our example). Set Hide Geometry for the pane packer to keep the wireframe clean and simple.

Duplicate the pane and move the copy to the right side of the first pane. Select both panes, and create a packer called top_row. Using the property window, enable Expand, Fill X and Fill Y for both child objects of the packer. Duplicate the top_row, move it below the top panes, then rename the copy as bottom_row. Select both rows and create a new packer. Rename it as 'panes'. Enable Expand, Fill X and Fill Y for its child objects (top and bottom rows). As before, hide the geometry of the created packers. Test the quad pane system by adjusting its width and height to, say, 0.6*0.8 meters. Then reset the size back to width=0.4 m, height=0.4 m.

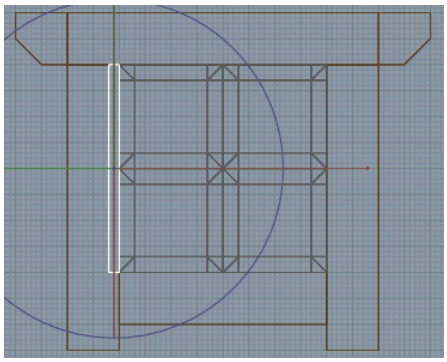
Window Edges

The next parts are simple to create. We will add 4 cubes (wooden boards) around the panes. The purpose of these cubes is to cover the edges of the window hole. Therefore, the board depth must be adjustable so that it can be matched with the thickness of the wall. In order to achieve desired z-direction packing, the board layer is split into two parts, as explained below.

The first part consisting of 4 packed cubes surrounds the panes, as shown in the picture below.



Take a front view and create a 2 cm wide cube. The height should match the height of the window panes (40 cm), and depth the depth of the pane profile (4 cm). Take a side view and move the cube so that it matches the side of the panes. The cube selected, activate the packer tool and accept it. Rename the packer as board_left. Deselect the board and switch to the top view. Go to the materials tab, make sure wood is still selected and activate the Parallel Map tool from the popup menu. Draw the mapping cube (5cm*5cm) around the board. Switch back to the geometry tab and drag and drop the parallel map into the new board_left packer. Make the parallel map WF-invisible. Select the board_left packer, open the property window, set Target=cubeX and set the Fill Y and Expand options.



First board added

Duplicate the board_left packer and move it to the opposite side of the panes. Rename it as board_right. Make two copies of the board_left, rotate them 90 degrees and move them above and below the panes. Rename the copies as board_top and board_bottom. Note that the top and bottom boards are not long enough yet, but the parent packers will later fix this.

Select board_left, panes and board_right. Activate the packer tool and accept. Set the packing options as follows:

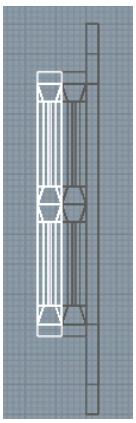
- Target=board_left: Expand=Off, Fill Y enabled
- Target=board_right: Expand=Off, Fill Y enabled
- Target=panes: Expand=On, Fill X and Fill Y enabled

Rename the new packer as mid_part.

Then select the board_top, mid_part and board_bottom packers. Apply MiscTools/Norm/Rotate to fix the object space orientation. Then create a new packer, which packs the three selected components. Rename it as wrapped_panes and set the packing options for its sub objects as follows:

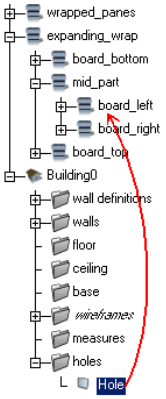
- Target=board_top: Expand=Off, Fill X enabled
- Target=board_bottom: Expand=Off, Fill X enabled
- Target=mid_part: Expand=On, Fill X and Fill Y enabled

Next we create the z-expandable layer of the wrapping boards. To make things simpler, just duplicate the wrapped_panes object and rename it as expanding_wrap. Move it behind the panes from the side view, so that it matches the other parts seamlessly.



wrapped_panes copied and moved

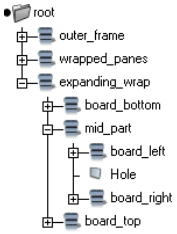
Open the hierarchy of the new duplicated packer expanding_wrap. Find the packer called panes and delete it. Removing sub objects of a packer usually changes the way how children are packed, so select the parent packer (called as mid_part) of the deleted panes and make sure that the children have correct packing options. Both boards should have Fill Y enabled, and Fill X and Expand disabled. Do not worry if the vertical boards move when you deleted the panes. We will next add a new component, which restores the shape.



If the window will be used with the building construction tools, it needs a hole cutting feature. Take a top view, activate the Building tool and create a straight wall behind the window. The direction should match the window. Click Accept to finish the building tool after the second point. Take a front view and activate the hole tool. Draw a hole, which matches the panes (40 cm * 40 cm). When the hole is added, explore the hierarchy of the building object to find the hole item (it is placed in a sub folder called holes). Drag and drop the hole to the hierarchy of the mid_part packer, between the board_left and board_right objects (that's the hierarchy position where panes were before we deleted it). You can also make the hole WF-invisible; it's wireframe does not show any information that is not already available from other parts of the window.

Select the new parent of the hole (packer mid_part), set Target=hole and enable Expand, Fill X and Fill Y. In other words, the hole fills the middle area of the expanding_wrap packer.

You can now delete the building object; it was just used for obtaining the hole. To save some work, you can naturally save a hole object, reload and scale it whenever you create new hole-cutting library objects.

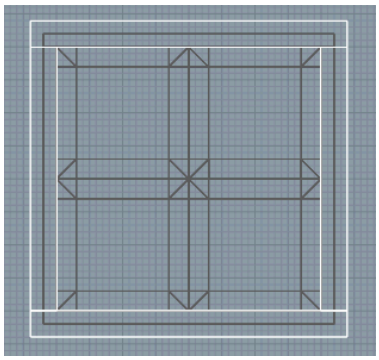


expanding_wrap packer hierarchy

Select the top level packer 'expanding_wrap'. On the property window, set Target=[All] and activate Fill Z option. Z filling is now used first time. After that, select each child packer from the select window and set the Fill Z option to all their sub objects. You should go trough all nested levels of packers and activate Fill Z for all children of all packers. Then try changing the depth of expanding_wrap packer from the current value 4 cm to say 10 cm. Make sure that all boards expand properly. If not, recheck that all Fill Z options are enabled. Finally, reset depth back to 4 cm.

Interior Mouldings

The last part of the window system is interior mouldings. First create four wooden boards, approximately 4 cm wide and 1 cm thick, which surround the window as in the example picture. Each board is a packer, which includes the geometry and the wood map. The geometry is packed using Fill X or Fill Y and Expand options. We do not repeat the creation steps here; they are identical with the steps for creating the edge wrapping boards above. In the example picture, the mouldings are joined in 90 degree angle. You can also use 45 degree beveled cuts between the mouldings, just like for the window panes. They are a bit more laborious to set up, though.



For the sake of clarity, rename the interior moulding items as moulding_left, moulding_right, moulding_top and moulding_bottom. Select moulding_left and moulding_right. Make sure the red object axis is horizontal when seen from the front view; if not, apply MiscTools/Norm/Rotate. Then create a new packer, which includes the two vertical mouldings. Rename the new packer as vertical_mouldings.

After the creation, the vertical mouldings jump slightly from their original positions. Open the property window and change the packing options as follows:

- Target=moulding_left: Expand and Fill Y enabled, Anchor=West
- Target=moulding_right: Expand and Fill Y enabled, Anchor=East

Now the mouldings return back to their correct positions.

Next select moulding_top, vertical_mouldings and moulding_bottom. Apply MiscTools/Norm/Rotate to fix the object spaces. Then create a new packer including these 3 components and rename it as mouldings. The packing options of the children are:

- Target=moulding_top: Expand off, Fill X on
- Target=moulding_bottom: Expand off, Fill X on
- Target=vertical_mouldings: Expand, Fill X and Fill Y enabled

Component Assembly

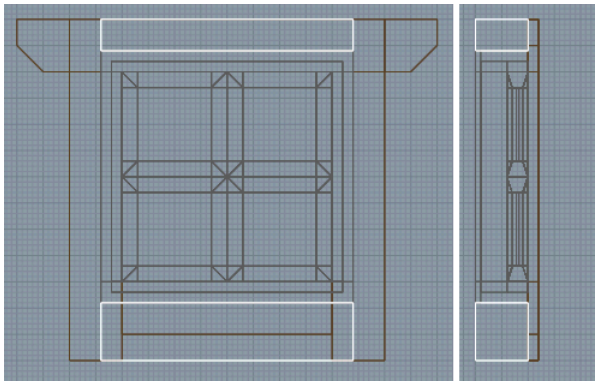
The window components are ready. We need to assemble them into a single packer. This phase requires some planning, because the components are not all symmetric. Simple centering is not enough - the outer frame must have certain offset values with respect to the other window.

First we pack the panes, the edge boards and the interior mouldings. These are symmetric and can be managed in a simple manner. Take a front view and select the three above-mentioned objects one by one. Object handles should be centered in the middle of the window (exactly in the middle crossing of the 4 panes). If not, drag the handle center (pivot point) into the center of the window. Take a side view and ensure that each component has its handles inside the geometry. The packer tool uses the pivot points to detect the direction of packing. Move the components so that they fit together seamlessly. Select all 3 parts (wrapped_panes, expanding_wrap, mouldings) and create a new packer. Rename it as window.

Set the following packing options for the children of the new window packer:

- Target=mouldings: Expand off, Fill X and Fill Y on
- Target=wrapped_panes: Expand off, Pad X = Pad Y = 0.02 meters, Fill X and Fill Y on
- Target=expanding_wrap: Pad X = Pad Y = 0.02 meters, Expand on, Fill X, Fill Y, Fill Z enabled

Next we pack the outer frame and the window together. To solve the asymmetric vertical positioning problem, take a front view and create two cubes. The first cube (rename it as 'fill cube1') should fill the gap between the top edge of the interior mouldings and the top edge of the outer frame. The second cube ('fill cube2') fills the respective area below the window; see the attached picture. Take a side view and make sure the cubes are aligned with the window packer. Decrease the depth of cubes so that they match the depth of the window packer (9 cm).



Two highlighted 'place holder' cubes define asymmetric packing offsets from edges

Select both cubes and make them WF- and RT-invisible using the MiscTools tab. The only purpose of the cubes is to reserve some space in packing.

Select the cubes and the window packer. Create a new packer and rename it as window. Set the packing options:

- Target = fill cube1: Expand off, Fill X and Fill Z may be enabled (filling is not relevant for invisible cubes)
- Target = fill cube2: Expand off, Fill X and Fill Z on
- Target = window: Expand on, Fill X, Fill Y, Fill Z on

The final packing in depth direction needs some special attention. So far, we have created packers using the automatic Detect option. The packer tool has figured out the packing arrangement and orientation itself. Next, we will guide the highest packing orientation manually, because we want to define, which point is the fixed anchor when dimensions change. For a door, the anchor point should be in the bottom front edge of the door. Then the door elevation will not change when the door height changes. For a window, the anchor point should probably be at the top, so that top edges of several windows remain aligned regardless of applied height changes.

Take a front view. Select the outer_frame first, then multi select the window. This time the selection order is important! Start the packer tool. Do not click Accept right away. Instead, change Orientation control in the toolbar to Z (we did not bother lining up the pivot points above). Click first time on the top left corner of the whole window object. Then move the mouse down and to the right, until the packer box matches exactly the width and height of the window. Click second time to finish the packer tool. The new packer has its anchor set to the top left corner of the view, as the XY arrows indicate. Hide the control box of the packer from the Wire tab's Hide Geometry option of the property window.

Rename the new packer as window; this will be the final window object. Pack the children objects as follows:

- Target = outer_frame: Expand off, Fill X and Fill Y on
- Target = window: X Pad = 0.16 m, Expand on, Fill X, Fill Y and Fill Z on

The parametric structure of the window is now ready.

Drag and Drop Settings

The final step of the tutorial is to set convenient properties for the window object considering its later use in house construction projects.

Select the complete window object (highest packer level). Make sure that its sub hierarchy folder is closed in the select window. This ensures that you can select the complete window object simply by clicking its wireframe on a view window. If the sub hierarchy is opened, view selections would activate sub objects, which is inconvenient considering possible transformations to new positions.

Open the property window. Go to its leftmost Gen tab. Set Hit-Invisible option for the window. This will make snap operations in complex models - with tens or even hundreds of windows - faster.



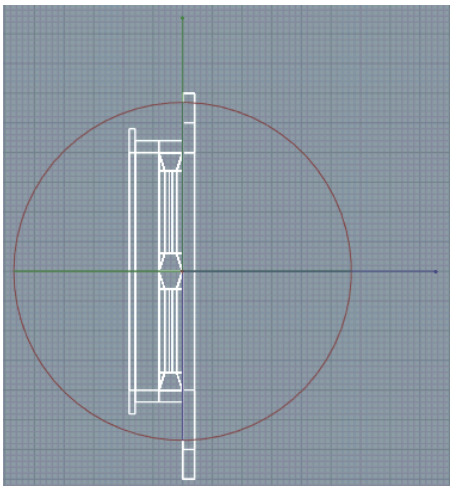
Note

Objects, which snap together (for example kitchen cabinets) should not be totally Hit invisible. However, their internal parts, such as shelves and minor details (handles, decorations) can be made Hit invisible to improve performance.

Switch to the Spec tab in the property window. Open the Drag and Drop Snapping frame. Here you can define what happens when the window object is dropped from the object browser to a view. The following options are suitable, when the window object handles are aligned with the world space. You can check the alignment by taking a front view while the window is selected. Red axis should point to the right, green axis up and blue axis towards the viewer. If the space is not like this, apply the MiscTools/Norm/Rotate to fix it.

- Space = Object. Snap targets are searched in directions defined by the window geometry.
- Dir1 = XZ. Window snaps to the nearest wall in XZ direction (horizontally). It does not snap to roof or floor.
- Dir2, Dir3 = None. Secondary snap directions can be inactive. These options are usually activated for corner searching. Our general purpose window object is not corner specific.
- Rotate=H,P. Object is first rotated around its y axis (Heading angle) to match the snap surface. Then it is rotated by its x axis (Pitching angle). Walls are usually vertical, so pure H (Heading) rotation would be appropriate for most cases, too.
- Axis=Z. The positive Z axis of the object will be rotated to match the surface normal of the wall. In our window, positive Z axis corresponds to the outer side of the window. Therefore, the outer window frame is rotated to the drop side of the wall. With these settings, the window should always be dropped outside the wall.
- Fit=Pivot. The object is moved so that its pivot point matches the snap position.
- Range=2 meters. This is a suitable finite snapping range. You can temporarily disable snapping by dropping objects farther away from the walls than this limit. On the other hand, it is easy to drop the window quickly within this snap range from a desired target wall.

Take a side view and make sure that pivot point is placed between the outer frame and the rest of the window - see the picture below. If not, drag the pivot point there. From the front view, drag the pivot point to the center of the window. Move the window object up/down so that the window is placed vertically to a suitable default height. For example, the pivot point can be 1.5 meters above the ground level. The elevation of windows in each building project will naturally depend on the wall and base heights, but this kind of default elevation ensures that wall-window snapping will work. Windows, which have been quickly drop-snapped to walls, can later be moved up or down.



Place the pivot behind the outer frame

Set the default size of the window using the dimension controls (Width/Height/Depth) of the toolbar. The window was modeled above to a minimal size w=0.8m h=0.65m d=0.11 m. A more common size might be 1.2*1.2*0.23 meters.



Note

The dimensions include the mouldings. If you want to control the size using the actual hole dimensions, drop the window into a new packer level and add some negative X and Y padding, which compensates the mouldings.

We can also optimize the real-time drawing of the window. When shaded OpenGL mode is enabled, the program computes up to several hundreds of pixels wide preview texture for each component of the window. The window object we created is actually not that simple; it includes almost 100 sub objects! Accurate preview computation will take long time when several tens of windows have been added to the scene.

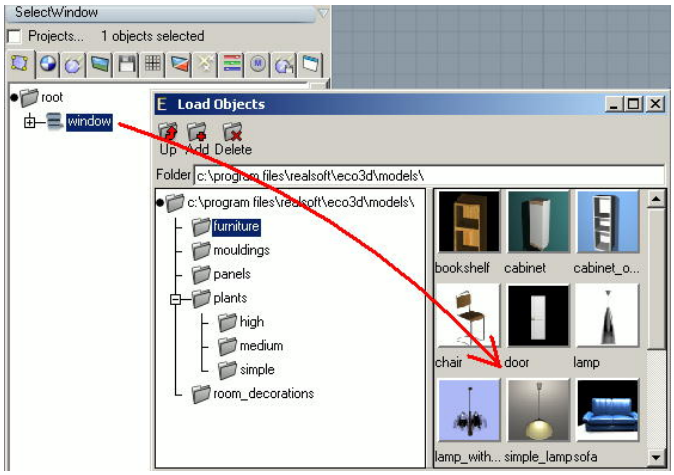
We can conveniently change all objects using the powerful advanced selection tool. The tool is available below the select window; click the toolbar to open it. First clear all selection by hitting the Clear button. Switch to the Class tab of the Advanced Selection window. It shows all object types included in the current scene. Find Polyhedron from the tree and click it to activate the class. Then click the Select button to select all polyhedrons of the window. Next activate Rectangle from the class tree and click Select. Then activate Cube class and click Select. Finally activate Nurbs Mesh class and click Select. Now all geometric surface items included in the window object have been selected. Click the Advanced Selector bar to shrink the window back to its original size.

Go to the property window's Wire tab. Change the texture Quality to a low value, for example 8. You can also reduce Shaded Quality to 1 or 2 to achieve even faster OpenGL drawing.

Saving with a Thumbnail and Materials

Let's create a nice thumbnail image for the window. Take a side view, zoom out and create a point light source above and in front of the window. Navigate the view to find a nice viewing position, which shows the shape of the window well (maybe a direct front view is OK). Zoom in so that window fills most of the view. Hit zero hotkey '0' to render the view. Open File/Save As window and click the Thumbnail image square to record the image from the view. You can also fill in the author and the comment information.

Close the Save Dialog without saving. Instead, open the File Browser window. Select a suitable target folder for saving in the browser window. Then drag the window object from the select window and drop it to the browser's object thumbnail area at the right side of the window. Drag and drop to the browser window saves the dropped object with the current project thumbnail and all materials that were used in the object. It is a quick way to save all relevant object information.

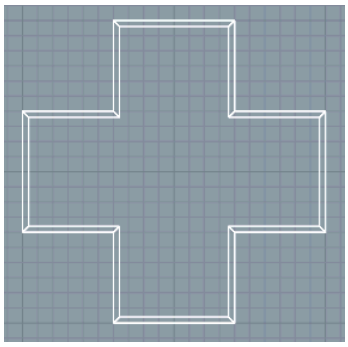


You can now find the created object from the browser window and start using it in your house construction projects.

Modeling Roofs

Multiple Roof Lines

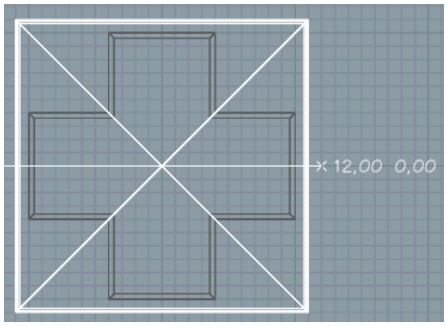
The roof object can be controlled using several roof lines. To try this, take a top view, activate the building tool and draw the walls as shown in the picture.



Draw cross-like walls

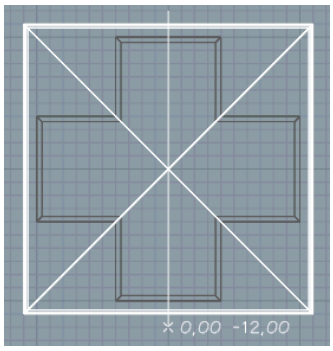
After drawing the walls, hit the roof tool button. It appears to the second row of the toolbar automatically when a building object is selected. The roof tool adds a default roof to the building. In this case, the default roof is a pyramid-like symmetric hipped roof.

The roof becomes automatically selected, and its context tool Roof Line appears to the tool bar. Click the Roof Line tool. From the top view, draw a straight line across the building as shown in the picture. Click Accept or hit the Enter key after clicking the second point.



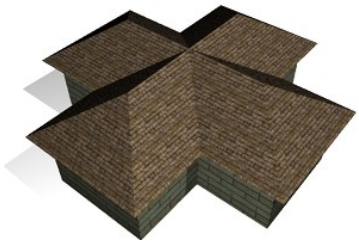
The first roof line

Note that the first roof line does not change the roof at all. It is still a pyramid like hipped roof. Activate the roof line tool again and draw a second line, which is perpendicular to the first roof line, across the building.



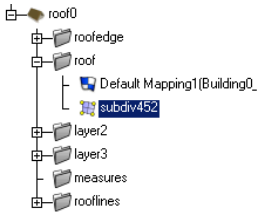
The second roof line

The two guide lines together define a cross-like roof.



Editing the Roof Manually

Automatic roof builder provides a quick way to add the roof. However, it is quite limited. The automatic roof object can be used as a starting point for modeling unusual roof shapes.

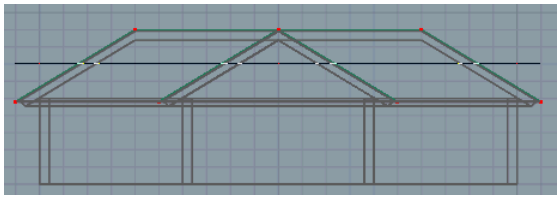


The roof surface selected

Select the roof object created in the example above. Open the property window (using 'p' hotkey or from the popup menu). Deactivate the Build option. Automatic roof construction is now disabled, and you can start editing the shape.

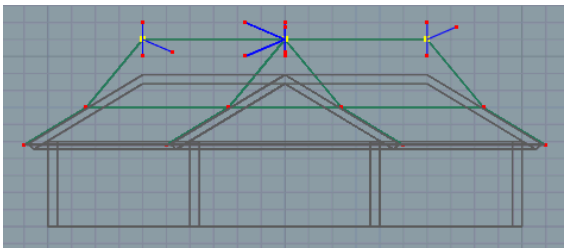
Select the object called 'subdivXX' in the sub folder called 'roof' of the roof creator object.

The second toolbar row shows an Edit mode control; set it to the Point mode. Then scroll the second row of the toolbar until you find the Knife tool. Click the tool to activate it. Take a front view and draw a horizontal line across the roof as in the picture below. Hit the Enter key after clicking the second point.



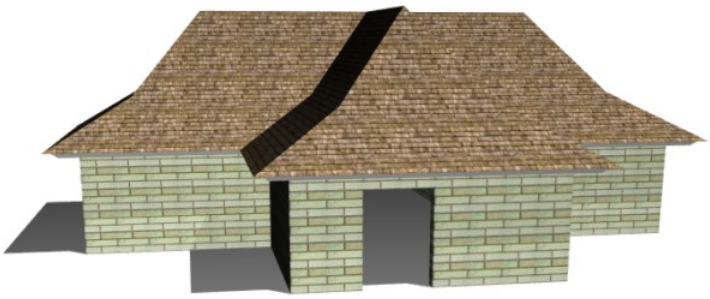
Cutting the roof top with the knife tool

Select the topmost points of the roof by LMB-dragging a selection box around the roof top. Move the pointer exactly on top of a selected point handle, press the left mouse button down and drag the points upwards about one meter. Be careful not to drag the blue handle lines - it will break the roof!



Move the roof top upwards

The modified roof is now ready. To improve rendering quality, open the UV editor from the Windows pull down menu and move the selected points further away from other points to compensate texture stretching, which appeared when the roof surface was stretched upwards.



Oriental roof style



Note

Enabling the roof's Build option removes all manually edited changes from the roof.

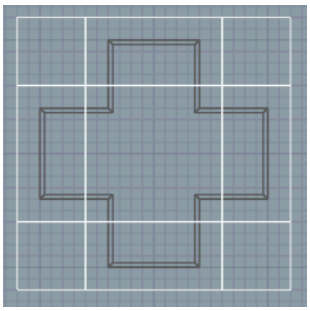
Custom Roof Shapes

The whole roof surface can be modeled manually. Placing the custom roof surface into the hierarchy of the roof builder object allows some automatic parametric features such as wall fitting with the roof.

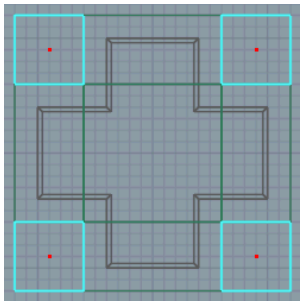
We will continue from the tutorial above. If you experiment with another building project, just make sure that Build option of the roof object is turned off. Select the sub folders roof, layer2 and layer3. Hit the del key to delete them. This removes all roof surfaces.

Take a top view. Activate the SDS/Rectangle tool from the toolbar. Draw a rectangle that is about 1 meter wider than the walls. From the front view, move the rectangle above the building.

Go back to the top view. Apply the SDS Knife tool 4 times to cut new sections through the SDS rectangle, as shown in the example picture.



SDS rectangle forms the roof



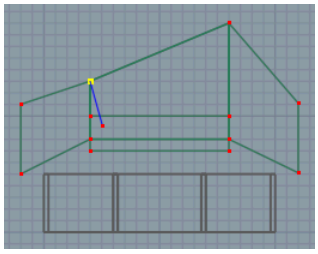
Corner faces selected

Set the Edit mode selector in the toolbar to Face. Click a corner face and then shift click the three other corner faces to multi select them.

Hit the Destroy tool button in the toolbar to remove the corner faces.

Set the edit mode selector to Point. Select the SDS points one by one from the top view, and apply the move tool (m hotkey) from the side and front views to move the points up and down, so that the SDS surface gets an irregular bumpy shape. Note that holding the y modifier key down while moving keeps the roof points on their original vertical axis.

Set the Edit mode back to Face. Apply the Dupl tool (Duplicate Faces). Hit m hotkey, press y modifier down and move the duplicated faces slightly upwards. The roof gets some thickness.



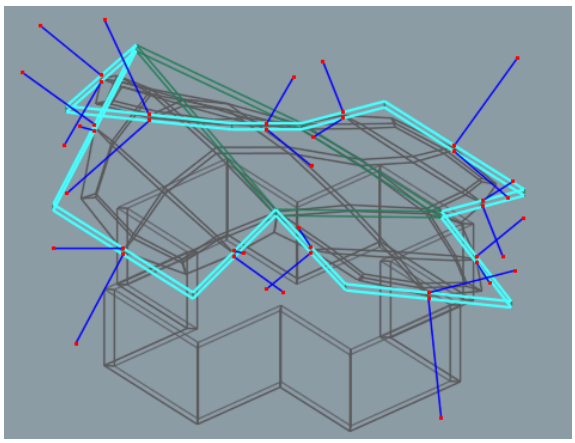
Roof points edited vertically

The top layer of the roof is now suitably selected. Take a top view and zoom in heavily so that you see only a small part of the roof (one texture tile). Switch to the materials tab and drag and drop the material buildingX_roof to the view window. This assigns it to the selected faces. Open the property window and select a suitable texture map for the buildingX_roof material.

Switch back to the geometry tab of the select window. Go to the spec tab f the property window and change the Rendering/Type of the SDS object to Smoothen to NURBS. The roof gets an organic rounded shape.

Set the SDS Edit mode to Edge. Multi select outermost edges of the top and bottom surfaces of the roof - see the picture below. You can do this easily from the top view. Just keep the Shift key down and drag small selection boxes which hit the desired edges. Do not select the short vertical edges at roof corners.

Set the rounding selector at the right end of the second row of the toolbar from Round to Sharp. You can make the edges sharp also using the controls available in the Spec tab of the SDS property window.



The roof edge loops selected

The roof is now ready. Set the Edit mode to Object and drag and drop the SDS object from the root level into the hierarchy of the roof object. It can be placed anywhere in the roof's sub hierarchy.

Select the Wall0 object in the wall definitions folder of the Building object. Activate Extend to Roof option from the toolbar. Walls expand up to the roof. To ensure accurate match, change the fitting density of the wall object to 0.1 meters from the toolbar.



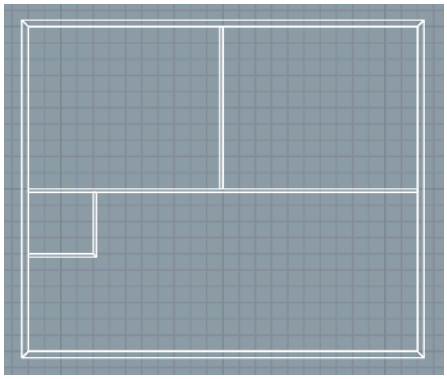
Gaudian roof style?

Multiple Floors

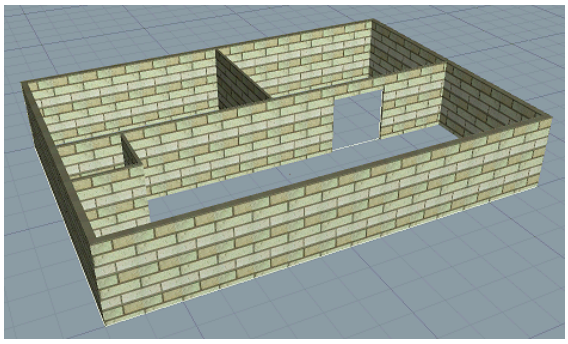
This tutorial shows how to create buildings, which have multiple floors.

Ground Floor

Start the building tool, take a top view and draw a building shown in the example picture. The dimensions of the rectangular building are 12*8 meters. When the outer wall is done, split the interior first with a horizontal sub wall, and then using a vertical sub wall. The third sub wall separates a 2*2 meter corner from the largest interior space. Cut a hole between the room at the top left corner and the largest room using the Hole tool. Rename the building object as 'ground floor'.



The walls of the ground floor



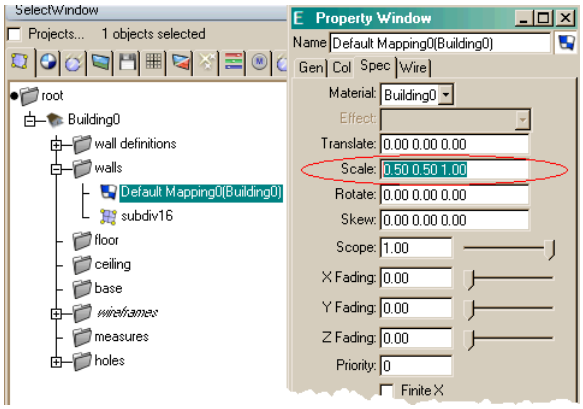
Too large tiles

Go to the second materials tab of the select window. Select the material building0 and open the property window. Select textures/tiles to the FileName field using the file browser.

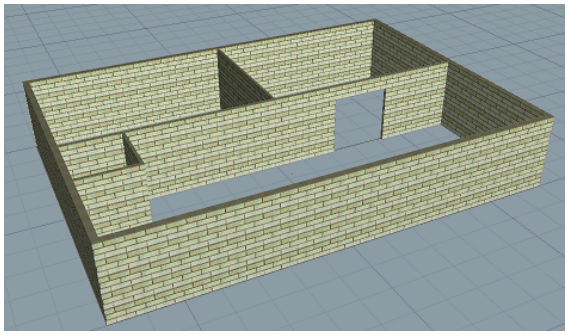
Go to the rightmost view models tab of the select window. Drop OpenGL shaded to the view window to see the building in shaded mode. The brick pattern is too large, because the selected texture includes only a few tiles.

The size is easy to fix. Go to the geometry tab of the select window. Select the item ground floor/ walls/default mapping(Building0). Open the property window and go to the spec tab. Change the Scale value from (1 1 1) to (0.5 0.5 1).

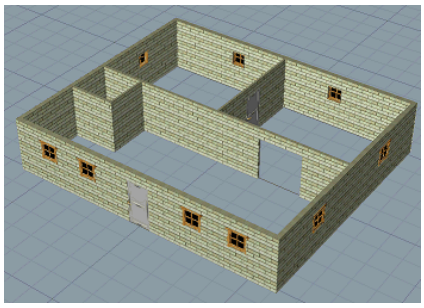
The tile size is reduced to 50 % from the original size, and now it is quite realistic. Close the property window.



Adjusting the texture mapping scale



Tile size fixed

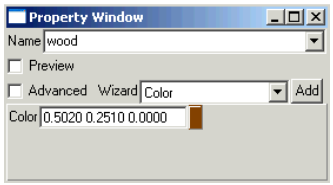


Doors and windows added for the ground floor

Select the ground floor (=the building object) and make it the current level using the Make Current popup menu of the select window. Open the Browse Object Library window from the popup menu. Drag and drop doors and windows for the ground floor. Restore the current level back to the root level.

Add a light source to the scene, navigate the view to see the building from a suitable angle and hit the zero key to render the view. Pinewood windows may not look that good in a brick wall. Go to the second tab of the select window to examine the material library. Select the material called wood. Then hit the del key to delete it.

Select 'New/VSL Material' from the popup menu. Open the property window and rename the material as wood. Set the Wizard selector of the property window to Color and click the Add button. A new color control appears to the property window; change it to a dark brown tone (0.5 0.25 0). The new kind of painted brown wood is now applied to all windows.

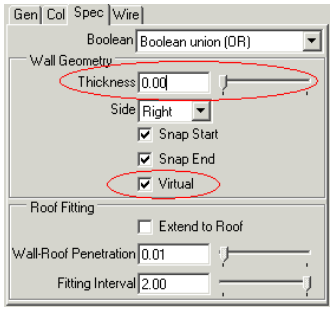


Pine wood replaced with brown painted wood

Select again the complete building (ground floor). From the top view, start the room tool. In the toolbar, define the name 'livingroom' and then click in the middle of the largest room. When the room decoration appears, turn Ceiling temporarily off to see into the room. Go to the materias tab and define the texture file names for livingroom_walls and livingroom_floor, for example 'wood' and 'plates'.

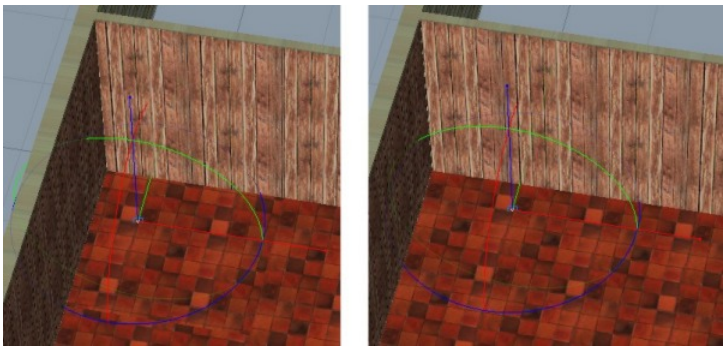
Select the livingroom object. Hit Ctrl-D to duplicate it. In the view window, move its position indicator into the small corner room. The corner room gets the same material attributes. Rename the copy of the room object as 'staircase'.

In an ordinary house, the staircase is not enclosed into a room. To remove the wall, select the last sub wall object from the wall definitions folder of the ground floor building. Open the property window, and enable the Virtual option. Then set the Thickness of the wall to zero.

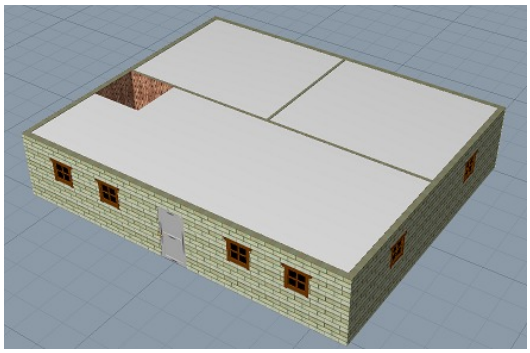


Properties of a virtual wall

The room tool sets the texture pattern relative to the position of the room indicator. Take a top view and make sure shaded OpenGL drawing is enabled. At this point, the texture pattern of the floor may not run continuously from the staircase area to the living room. If you see a visible seam, select the staircase room object. Then turn the grid snapping off and activate the move tool. Move the room indicator to find a position, which makes the pattern to run seamlessly. Turn grid snapping back on.



Move the room indicator to fix the texture seam



The ground floor finished

The corner room has apparently disappeared. However, it still divides the space. We can, for example, turn off the ceiling option from the staircase to cut a hole from the ground floor to the upper floor. Such a hole is a parametric feature and its shape and size can be edited at any stage of the building construction.

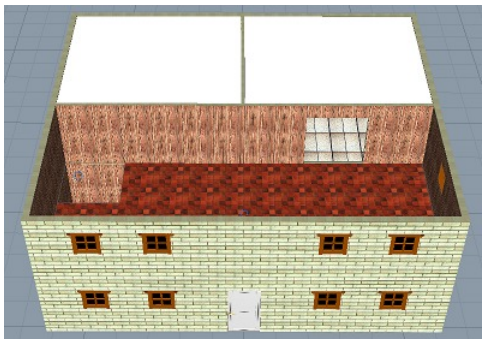
Apply the room tool two times more, to texture the remaining rooms. The rooms can be called, for example, as bedroom and kitchen. After texturing all rooms, enable the ceiling option for all rooms except the staircase. The ground floor is now ready.

Upper Floor

Select Edit/Duplicate from the pull down menu. Rename the duplicate as 'upper floor'. Move the upper floor upwards until it is placed exactly on top of the ground floor.

The upper floor obviously needs some editing. First of all, find the door object, which cuts the outer wall, and delete it from the hierarchy of the upper floor. You may also rearrange the rooms if you like, or add or delete some windows, or change some textures. Do not change or remove the staircase object.

Select the staircase object of the upper floor and open the property window. Staircase downstairs had ceiling disabled and floor enabled. To finish the hole between the two floors, disable the floor option for the upper staircase room. You can now see all the way downstairs through the hole.

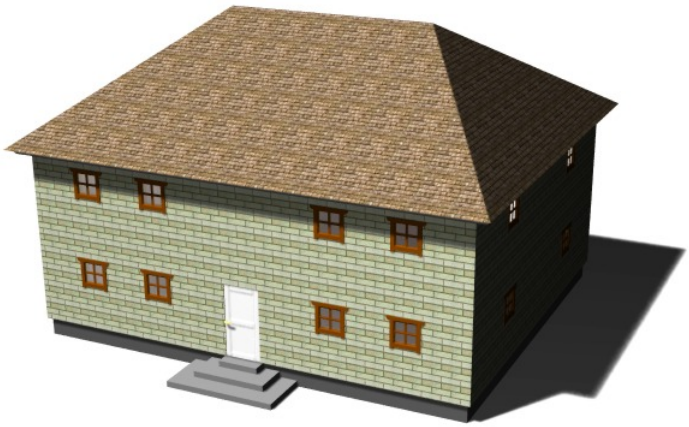


A hole between two floors

If you zoom in and look closely in shaded OpenGL mode, you can see a narrow gap between the wall textures of the upper and lower staircase rooms. To fix it, select the staircase object in the upper floor and using the property window, set its floor offset to zero. Then select the staircase object under the hierarchy of the ground floor and set its ceiling offset to zero. The gap disappears.

The actual stairs can be modeled using the general modeling toolset of the software. With the macro feature, it is easy to build repeating forms. The steps can be cut to match the staircase area using the Boolean tools. For details, see other modeling related tutorials of this manual.

To complete the building, enable the ceiling option for all rooms of the upper floor. Select the upper floor object and apply the roof tool to it. Set desired roof options using the property window. Go to the material tab of the select window, select the upperfloor_roof material and set a suitable roof texture for it using the property window. Select the ground floor. Set its Base type to Fitted to Walls using the selector in the tool bar. Model the stairs at the front door for example using analytic cubes.



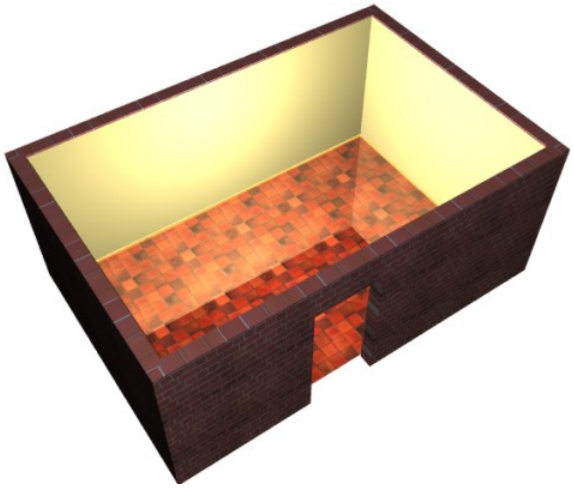
The finished building

Designing Kitchens

The packer object is a very useful tool for managing furniture groups. The next tutorial about kitchen design is a good example.

Take a top view. Start the Building tool and draw a simple rectangular building. Then start the room tool. Name the room as kitchen and click in the middle of the building to add the room. Switch to the materials tab and assign suitable texture maps to the kitchen floor, wall and the ceiling materials using the property window. Create a point light in the middle of the kitchen, 0.3 meters below the ceiling.

Go to the grids tab of the select window. Drag and drop a 0.1-meter grid to the view window. It is a suitable grid for kitchen design.



Empty kitchen

Switch back to the geometry tab of the select window. Take a top view. Open the Browse Object Library window from the popup menu of the select window. Drag and drop the object 'Cabinet' from the furniture folder into the view window, close to the top left corner of the back wall. Drop the cabinet closer to the back wall than the side wall, so that it snaps primarily to the back wall. If the drop position was properly chosen, the cabinet ends up in the corner as in the example picture.



Drop a cabinet to the corner

Click the Packer tool of the Creation tab of the toolbar. Click Accept to finish the tool with the default options. The new packer takes its orientation from the selected cabinet. Rename the packer as 'wallcabinets'.

While the packer is selected, hit the Tab key or apply Make Current from the popup menu of the select window. Then new cabinets will be inserted automatically to the packer.

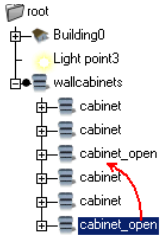
Now you do not need to aim at all when inserting new cabinets. Just double click the item Cabinet again in the object browser window. The new cabinet goes automatically to the end of the row.

The loaded cabinet is automatically selected. Change its width to 0.6 meters using the tool bar controls.

Double click the icon 'Cabinet_open' on the object browser window. Change the width of the new cabinet to 0.8 meters.

Double click the Cabinet icon 2 times more, to add two new cabinets to the row.

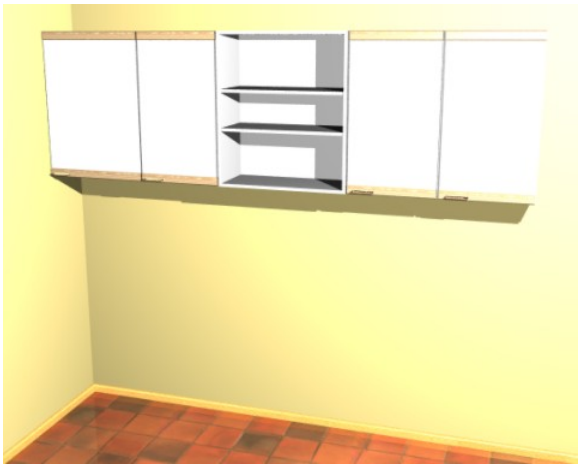
Select the second cabinet. Change its width back to 0.4 meters. Cabinets move to fill the gap.



Changing the order of cabinets

Double click the 'Cabinet_open' icon to load it. As expected, the new cabinet goes to the end of the row. In the select window, drag the last cabinet in the list (=the new Cabinet_open) and drop it just below the other open cabinet in hierarchy. The open cabinets will be placed side by side.

Select the new open cabinet. Hit del key to delete it. Cabinets move together to fill the gap.



Wall cabinet row finished



Chapter 7. Materials

Introduction

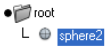
When creating photorealistic images and animations, it is necessary not only to create complex shapes, but also to simulate different real world material properties. Realsoft 3D includes powerful tools for this purpose.

Mapping materials to objects using drag and drop methods

The fastest and easiest way to assign a material to a geometric object is to drag and drop. Select the target object and drag a material from the material library, and drop it on the view window. The next example demonstrates this straightforward technique.

1. Select Sphere from the Analytic toolbar and create a sphere by clicking twice on the view window.

2. The sphere tool leaves the newly created object selected. If you were assigning a material to a previously created object, you should select it now. The selected object will be the target of the dropped material.

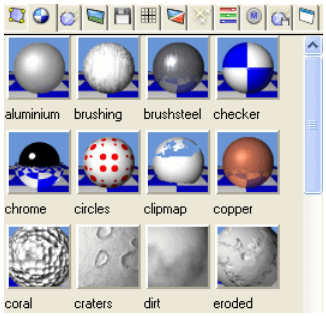


Sphere selected

3. Go to the second tab of the select window, called Materials. This will display the contents of the current material library.

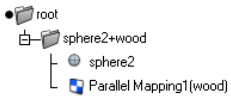
4. Move the mouse on top of the material 'wood', press the left mouse button down, then while holding the button down, move the mouse over the view window. Release the left mouse button there. In other words, drag&drop the wood material onto the view window.

That's it. Note that the position on the view window where you release the left mouse button does not matter. It does not select the target object, because this would be too inaccurate.

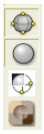


The material tab of the select window

Switch back to the object hierarchy section of the select window. It should now show you an object named 'Sphere+wood'. Open the sub hierarchy of that level by clicking the small '+' box at the left side of the level 'Sphere+wood'. The level object consists of two sub objects: the original sphere and wood. In other words, you have a wooden sphere. Realsoft 3D combines shapes and materials using the powerful and intuitive concept of hierarchy. Thanks to this, you can add new parts to the geometry later and they automatically get the right kind of (hierarchically defined) material properties.



The structure of a wooden sphere in the select window



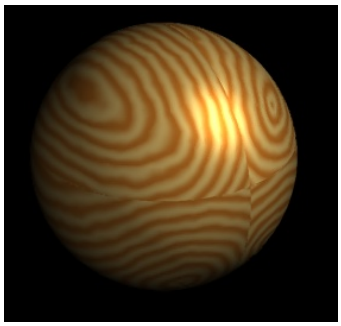
View control bar

The new material link object 'Parallel mapping(wood)' is called the *material mapping object*. The material to which the mapping refers affects all the objects in the same hierarchy level with the mapping object.

Let's render the sphere to see if the material works:

5. Use the Point light tool of the Light Sources section of the toolbar to insert a light source above the sphere.

6. Click the Render button in the View Control Bar to render the sphere (or use Render/Ray trace from the view's popup menu).



The wooden sphere rendered

Mapping multiple materials to a single object

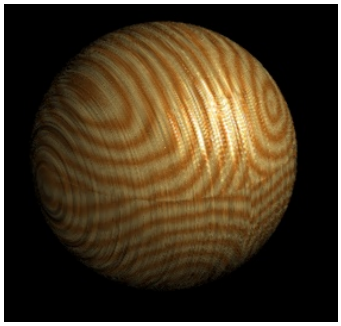
Attaching any number of materials to a single object is just as easy. For example, to turn the wooden sphere to a shiny grooved wooden sphere:

1. Make sure the 'sphere+wood' level is still selected in the select window's geometry tab. Note that the previous material dropping automatically leaves the hierarchy to the correct state.

2. Drag & drop the 'plastic' material into the view window.

3. Drag & drop the 'brushing' material into the view window.

4. Render the view window.



Grooved shiny wood

The plastic property makes the surface shiny, and brushing adds small irregular grooves to it.

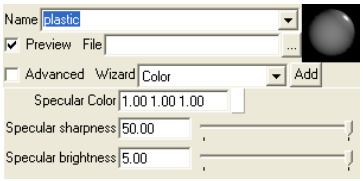
Modifying material properties

Most materials of Realsoft 3D, including wood, plastic and brushing, are not fixed but allow various adjustments. Let's change the shininess of the modelled sphere.

1. Go to the Materials tab of the select window. Click the material plastic with the left mouse button to select it.

2. Select Properties from the select window's popup menu. You can also combine steps 1 and 2 by double clicking the 'plastic' material icon.

3. The property window opens showing the material's properties. There are only two sliders!



Properties of plastic material

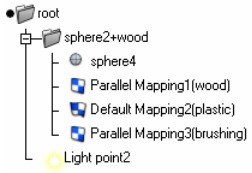
4. Decrease the Specular brightness value from 5 to 2 and Specular sharpness from 50 to 10. Close the Property window.

5. Render the view window. The shiny area is now less bright but wider, because the sharpness was reduced. The material combination looks more like varnished wood now.

One might think that the material editor of Realsoft 3D is poor, because there are only two sliders available! The truth is quite the opposite. What we see here is a tailored interface for the plastic material, which is so simple that anybody can change the material right away. Instead of displaying hundreds of confusing buttons and sliders, the program only shows the characteristic properties of each material. Later we will see that there are other, more advanced and powerful levels of control available, which can be adopted step by step as you become familiar with the program.

Modifying mapping objects

The hierarchy of the model is now like this:



Many materials assigned to a sphere

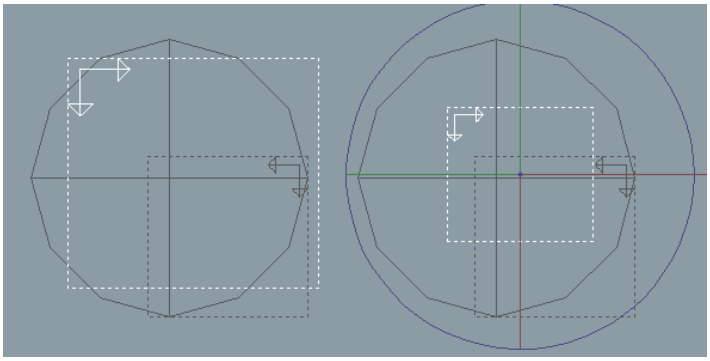
As we have demonstrated in this tutorial, Realsoft 3D uses geometric mapping objects in the scene hierarchy to assign materials. One great advantage of this approach is that modifying a material's geometric properties such as position and density becomes very easy and intuitive. Also operations such as duplicating, cutting and pasting materials or combinations of materials to affect new objects can be done efficiently.

For example, let's try to make the grooving of the wooden surface denser:

1. Select the object 'Parallel Mapping(brushing)' from the Select window.

2. Go to the Transformation tab of the toolbar and activate the Scale tool. This tool is for resizing objects.

3. Press the Shift key (Shift modified produces uniform scaling so that the proportions of the mapping will not change), and click in the middle of the selected parallel mapping object. Then click on the corner of the cube (while still holding shift down) and move the mouse towards where you first clicked. The parallel map shrinks accordingly. When the size has halved, click a third time to finish shrinking it.



Shrinking the groove map

4. Render. The grooves are twice as dense.

Similarly, you can use the rotate tool (or object handles) to change the orientation of the grooves. The same tools, which you use for building your models, can be used for manipulating material patterns and bump maps. Furthermore, you can see the result in a visual form in the view window, without any need to look for material specific 'density' sliders, etc. somewhere. This saves you from having to learn specific material positioning tools.

The drag & drop method presented above provides a very fast way to assign materials to objects. For the materials above, it is the most suitable way. However, sometimes drag&drop does not provide enough control

of how the material assignment should take place. For example, if we now want to paint a photograph of a logotype onto the wooden sphere, the program can't know if the logotype should cover the whole surface (wrapped around the sphere), or if it should occupy just a small rectangular area on top of the sphere. In such cases, it is best to use the actual mapping tools included in Realsoft 3D. However, before presenting these tools, let's examine how to create a new material.

Creating a new material

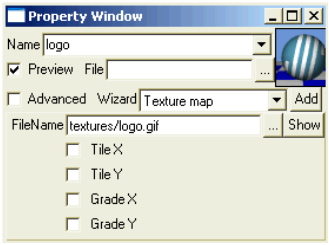
The next task is to add a small logotype on the north pole of the sphere. You can draw the logotype with a paintbrush program, or simply use any example image in the included textures folder. Using a bitmap image for painting surfaces is a very common procedure in 3D graphics. It's usually called 'texture mapping'.

1. Click the right mouse button in the Material tab of the Select window and select New/From Template from the opened pop-up menu. A file browser appears, showing the contents of the Materials folder of the program installation. Go to the Image Maps sub folder and select the item called Color Map. Click Open. A new material is inserted at the bottom of the Select window.

2. If the property window is closed, double click on the newly created material icon to open it. Change the material name from 'Color Map' to 'logo' using the uppermost Name field.

3. The property window shows some texture map specific controls. Click the browse button labeled as '...' beside the File name gadget and find the image (for example, pick the image 'logo' from the textures folder).

4. Set the Preview check box on the material editor. A small view window for previewing the current material appears. You see the result of rendering immediately. In addition, the preview image in the select window is updated. The texture map material is now ready for use, and you can close the property window.



A texture map material

The material collection, which you can see in the Select window, contains frequently needed materials. Some of the materials are ready to use ('Brushsteel' is such a material). Some of the materials define just one basic property, like a color pattern (wood, stripes, circles, checker, marble, granite, etc.), or bumps for the surface (wrinkles, waves, reptile, coral, craters, brushing), or optical properties (plastic, mirror, glass). You can combine these materials to get more interesting ones, just as we did earlier in this example.

In addition to materials of the default library, the materials folder contains many more useful materials, and you can insert these to the current library using the New/From Template tool. Some of the templates are empty skeletons like the color texture above, and you have to fill in the missing information. Others are ready to use.

Finally, the user can create almost any imaginable shading or material effect using VSL, *Visual Shading Language*. This powerful tool will be explained in detail in more advanced tutorials.

Using the mapping tools

Now we will insert the created texture map material to the scene using the accurate mapping tools:

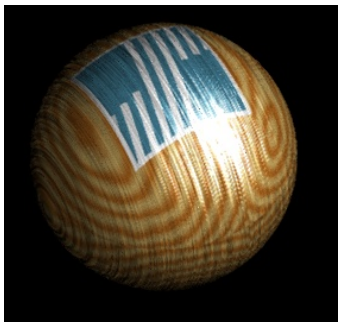
1. Go to the object hierarchy section of the select window and activate the level 'sphere+wood'

2. Take the top view (e.g. View popup menu/Camera/Top) so that the top half of the sphere is well visible.

3. Go to the material tab of the Select window and click the right mouse button over the logo material - a pop-up menu opens. Select the Map/ Parallel menu item. This activates the parallel map tool.

4. Enter two points through the view window to define a rectangle. The logo texture will be placed into this rectangle.

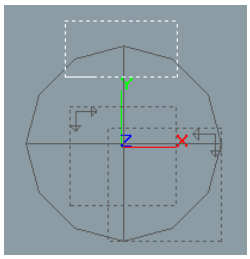
The mapping operation is now ready. Render the view to check the result.



The new texture map inserted

Rotate the camera downward or take the bottom view. Then render again. You should see the same logo image at the bottom of the sphere, but as a mirror image! This is because the parallel mapping works at an infinitely long distance by default. To change this:

1. Go to the Select window's Geometric Objects tab and select the 'Parallel map(logo)' mapping object.



Logo map encloses the top of the sphere

2. Take the front view and move the mapping object upwards using the object handles or the move tool. Move it so that the topmost part of the sphere is inside the mapping box.

3. Open the property window of the mapping object and go to the Spec tab. The property window shows you parallel mapping specific options, like the name of the mapped material.

4. Set the Finite Z option. This instructs the program to project the material only to the depth indicated by the mapping object's shape. Finite X and Finite Y do the same in the width and height directions, but they are not needed this time because the logo texture has a finite size and we did not activate the tiling option.

Now the logo map affects only the upper side of the sphere.

The last steps of this tutorial show you how to get rid of a material mapped to an object. To remove the brushed grooves from the sphere:

5. Go to the Geometric Objects tab of the Select window and select the 'Parallel map(brushing)' object.

6. Press the right mouse button and select Delete from the Select window's popup menu.



Note

The mapping object contains references to the materials, not the materials themselves. This approach has several advantages. Often you have to adjust the material properties afterwards. It is sufficient to modify the material just once to get the desired change in all objects which are made of the material. Secondly, a material is a large collection of information and therefore this method saves a lot of memory. The third advantage is that the same material can be used in many ways and with different kinds of objects. For example, a material can be wrapped nicely around a sphere using spherical mapping and spread over a flat plane using parallel mapping, without need to modify the material itself.

Blending Materials

This tutorial chapter introduces new ways of combining materials. You will learn how to mix materials:

- 50 % of wood
- 20 % of steel and
- 30 % of stone.

The tutorial also explains how to limit the effect of a material to an arbitrary area.

Level: Beginner/Medium

The chapter "Materials" explains how to assign multiple materials to a single object. Putting several materials into the same hierarchy level automatically combines those materials. This Tutorial will explain how the materials are combined and how you can control that blending.

The first three materials *wood*, *plastic* and *brushing* do not share a common property. *Wood* defines the color pattern, *plastic* the shininess and *brushing* the bumpiness of the surface. Only the fourth material *logo* includes another color pattern, which replaces the wood beneath it. This happens because *logo* was mapped to the object after *wood* and therefore is further down in the hierarchy.



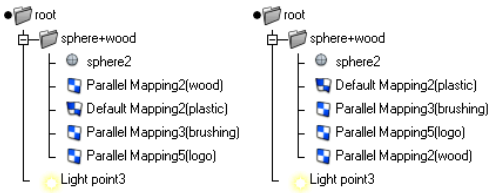
Note

Materials are evaluated in the same order in which their mapping appears in the hierarchy (from top down). A mapping applied later will be evaluated later and therefore it might overwrite all previously applied properties.

Now let us begin: Load the object, which you created in the previous chapter or the file "tutorprojects/material/mapping/multimap.r3d".Then do the following:

1. Render the view window to make sure that the logo is visible on the top of the sphere.

2. Place the mouse pointer above of the topmost mapping object Parallel Mapping7(wood); drag it over the icon of the lowest mapping object Parallel Mapping10(logo) and drop it when the cursor changes to an down arrow. This moves the wood mapping to the bottom of the list.



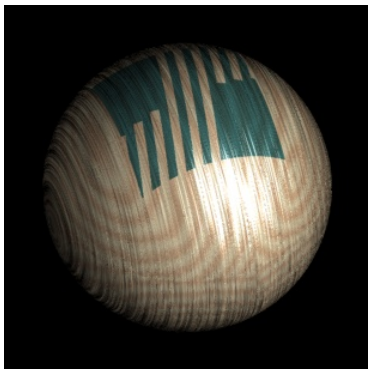
Logo is mapped after wood in the original hierarchy and replaces the wood in the area where it is defined. The hierarchy on the right side shows a situation where wood overwrites logo.

3. Render the view. Now wood grains are clearly visible also at the logo area.

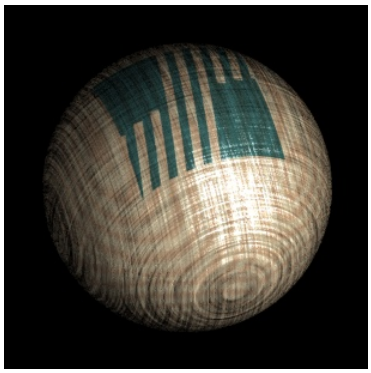
Now continue as follows:

4. Double click the wood mapping object to open the property window. Click the Spec tab to see the mapping specific options.

5. Set Scope to 0.5, close the property window and render the view. The wood material is only partly visible now and the other materials (white default material and logo on the top part) are visible through it.



50 % scope makes wood transparent and it is blended with previous materials



Combined brushing patterns

What happens if we add another bumpy material to the sphere?

6. Activate the object Parallel Mapping9(brushing) from the select window. In the front view, press the 'Ctrl' key, and simultaneously rotate the object - using the blue rotation handle - about 90 degrees. Now release the 'Ctrl' key. This is a quick way to create a rotated copy of the mapping. You should now see two copies of Parallel Mapping(brushing) in the select window.

Based on the previous experiments, you might now think that the new copy of brushing overrules the original one, because it comes later in the object hierarchy. Test this by rendering the view window. You will see two orthogonal brushing patterns, which is not what you might have expected.

This highlights that this bump map behaves differently from the color patterns. These bump maps are combined (added) whereas color maps overrule each other unless scope values are used to blend the materials in a desired fashion. The materials in the default library are deliberately constructed to follow this (useful and intuitive) convention. But please note! Realsoft 3D has an open architecture and you can easily implement any other kind of behavior using the material system. Bump maps can replace each other if required.

Local materials

Most materials in the previous example affect the target object all over. You may enlarge the sphere as much as you like, but you will not "run out" of wood. Only the logo material has a local influence at the top of the

sphere. The influence area is defined by the cube-like geometry of the mapping object and is locally limited because we activated the Finite setting for the parallel logo mapping.

It is often necessary to limit the influence of a material to a more arbitrary area than a simple cube. We will practice this by making a marble plate with some golden letters on it.

You might want to load the sample project: tutorprojects/material/mapping/goldtext.r3d

Tutorial level: Medium

1. Select Environments/Default Startup from the main menu or reload startup.r3d from the main Realsoft 3D folder to initialize your working environment and get the full material library back after the previous tutorial.

2. While in front view, create the plate. For example, use an analytic cube and rename it to 'plate'.



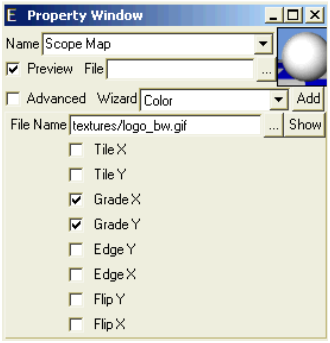
White areas will be made of gold

3. Start a paint program such as Paintbrush, which comes with the MS Windows Operating System. Make a black image with some white text on it and save it with a suitable name (e.g. 'textures/mytextimage').

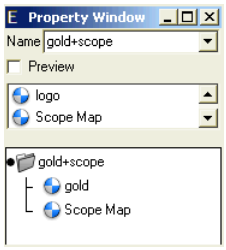
4. Make sure that the plate object is selected in Realsoft 3D. Go to the Materials tab of the Select window and drag&drop the marble material to the view window. This assigns the marble to the selected object.

5. Select New/From template from the material's select window's pop-up menu. Select the template Image Maps/Scope Map using the file browser and click Open.

6. Double click the icon of the new Scope Map material to open the property window. Set File Name to the text image which you created. Activate Grade X and Grade Y options. Grade options will blend the gold text smoothly to the background, thus reducing the jaggedness of pixel edges of letters.



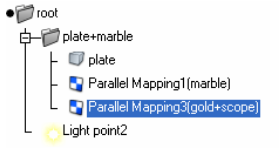
Properties of the scope map material



The level material includes two sub materials

7. Go back to the select window and select New/Level Material from the popup menu. A new material appears in the select window. Using the property window, rename it to 'gold+scope'. The upper area of the property window shows the contents of the material library and the lower area the contents of the created level material. Drag and drop both Gold and Scope map to the node on the lower part. Close the property window.

8. Make sure that the level containing the plate and the marble mapping is selected. Switch to the Materials tab of the select window. Move the mouse pointer over the icon of gold+scope material and choose Map/Parallel from the popup menu. Then create a mapping rectangle in the view window. The mapping rectangle should fit inside the plate; its size defines the size of the text.



Gold text mapped to the plate



Golden text in a frame

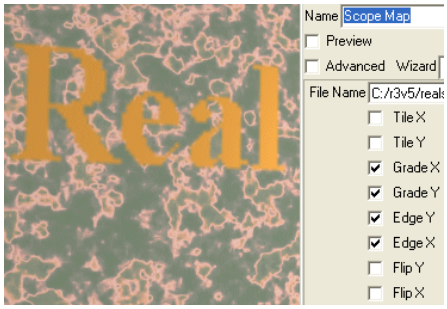
Render the view window. The result is indeed golden text on marble background, but there is a golden frame around the text as well.

The explanation for this is that the scope map cuts the influence of the gold only inside the black and white texture. Nothing limits the influence of gold outside the scope map. This is easy to fix:

9. Go to the Geometric Objects tab in the Select window and select Parallel mapping(gold+scope). Open the property window, switch to the Spec tab and set Finite X, Finite Y and Finite Z. You may recall that this limits the material to have influence only inside the mapping geometry. Close the property window.

10. Activate the side view and make sure that the selected gold+scope mapping object actually intersects the front side of the plate. Otherwise, because of the Finite options, the golden text would not appear on the plate.

11. Switch back to the front view and render. There is no gold at the edges of the plate, but if you look very carefully, you may still see a thin golden line following the edge of the parallel map. Why? Activate the Materials library on the select window. Double click the Scope Map material to open the property window. The Grade X&Y check boxes are set in order to smooth the scope image. These gradient options work also at the edges of the texture map, and the 0 level (no effect) scope of the background is blended to the surrounding level 1 (full) scope. Set Edge X and Edge Y options to disable the edge blending. Render the view to check the final result of the tutorial.



Set Edge options to remove thin gradient at the edges of the scope map

Instead of using Paintbrush for drawing the texture image, you can also use the 3D painting tools of Realsoft 3D to paint the scope mask image. For more information, read the tutorials about 3D painting.

Texture Mapping

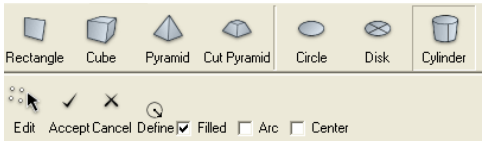
Attaching painted, scanned, etc. images to objects is one of the most commonly used techniques in texturing. This chapter examines the basic tools and controls for texture mapping.

Level: Beginner/Medium

Example file: 'tutorprojects\material\mapping\can'

In this tutorial we will create a soft drink can with some labels painted on it. For the sake of simplicity, the shape of the can may be just a simple cylinder.

1. Activate a top view and switch to the Analytic tab of the toolbar. Select Cylinder from the tool's menu. Then create the cylinder with two mouse clicks on the view window.

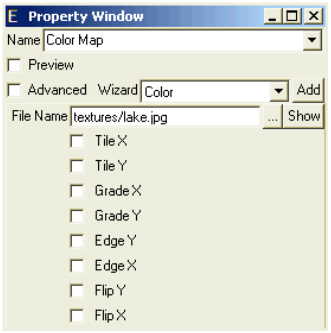


Modeling a can using the Analytic cylinder tool

2. Activate a front view and stretch the length of the cylinder until the shape is suitable for a soft drink can. To do this you can use:

- the scale handle at the end of the cylinder's handle axis
- the numerical axis control in the property window's Spec tab
- or the extend or scale tools

3. Go to the Materials tab of the Select window. Choose New/From Template from the popup menu. Using the file browser, select the template Image Maps/Color Map. Click Open.



The color texture map material

4. Double click the icon of new Color Map material to open the property window. Using the browse button at the right side of the File Name gadget, select the sample texture 'lake' from the textures sub-folder of the Realsoft 3D installation directory. You can also use any other image if you like.



Cylinder mapping

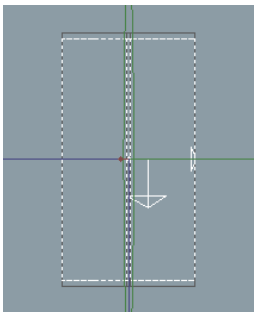
5. Close the property window. The cylinder should be selected at this point; if not, click its wireframe to select it. Then drag & drop the Color map material from the select window to the view window.

6. This drag & drop action automatically creates a cylinder mapping that matches the can. This happens because the Color Map template is defined to behave this way (find out more about preferred mappings from the VSL tutorials). The cylinder mapping wraps the selected image nicely around the can and is a good starting point for further editing.

7. Slightly rotate the view downward from the front view and render it.

As you see in the image, there are some problems. The bottom of the shows strange artifacts

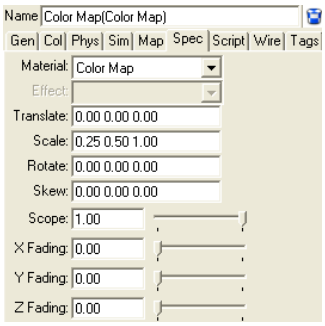
8. The artifacts at the bottom indicate accuracy problems. The cylinder map and the can have *exactly* the same length. It is impossible to say which one ends first, the texture mapping or the can (computers have only finite accuracy). Therefore, the rendering computations produce random results when trying to decide which one ends first. This is easy to fix. Scale the cylinder mapping to make it shorter using the scale handle and move it so that the mapping is in the middle of the can (see the image below). Render to verify that the mapping works well this time.





Cylinder mapping shortened and centered

9. The logo texture goes all the way around the can which makes visualizing the image hard. So let us make it smaller. Open the property window of the mapping object and go to the Spec tab (make sure that the mapping object is selected and the geometry objects tab is active on the select window). Change the Scale gadget value to '0.25 0.5 1'. Then render again. The width of the logo has become one quarter and the height one half of the original size.



Adjusting the mapping scale



10. Keep the property window open but switch to the Materials tab on the Select window. The property window shows now the controls of the Color Map material. Activate Tile X and render again. The logo pattern now is repeated and travels around the can. You may close the property window now.

11. The stripe of shrunk logo labels appears in the top half of the can, but how do you put it at the center of the can? You might have noticed that the mapping specific properties not only include Scale but also Translate', Rotate', etc controls. You can use the Translate gadget for accurate numeric positioning. However, the mapping geometry itself provides a more straightforward way to place the logo. Simply drag the cylinder mapping downward using its blue translate handle.



Tile X activated

Note that using the unified tools, you can create *a cylinder sector map:*

- Activate the unified circle tool from the pull down menu
- Set Class to Cylinder Map and select a material from the list in the tool control bar

- Activate the Arc option
- Draw the sector

Such a sector map allows intuitive control over the mapped area without the need to use numerical scale attributes.

Texture anti-aliasing

Let's experiment a bit more with the logo material created above.

Level: Beginner/Medium

Example file: tutorprojects\material\mapping\antialiasing

Select the Cylinder+Color Map from the object hierarchy and delete it.

Go to the top view and zoom out very far. Select Rectangle from the Analytic tools menu and draw a rectangle which covers the whole view.

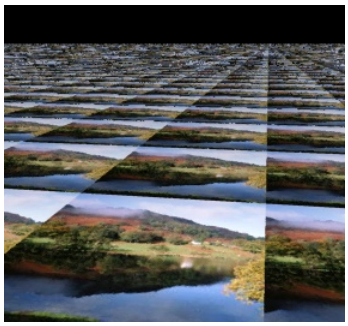
Zoom back in to a normal scale. Activate the Materials tab of the toolbar and choose the Parallel mapping tool. Select the Color Map material and draw a parallel map in the view window. The size of the map should be much smaller than the size of the rectangle.

Set the view back to the front view. Go to the Light Sources tab and activate the Ambient light tool. Click Accept. Now the simple test scene is ready.

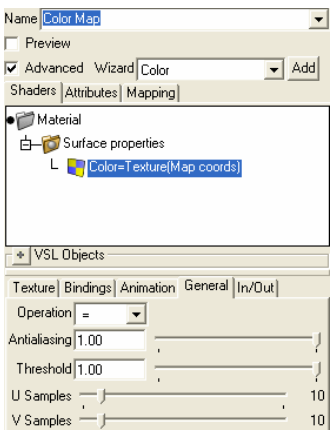
Go to the Materials section of the select window, activate the Color Map material created in the previous tutorial and open its property window. Tile X is already set but set Tile Y as well.

If the view window is in parallel projection, select Camera/Perspective from the view's popup menu. Rotate the view window upwards and render. The result should look like in the example image.

The logo can be seen well in the front area, but the quality gets worse as the distance grows. Improving rendering quality helps a bit, but any reasonable quality cannot completely remove problems appearing far away from the camera. It is better to modify the material itself to fix this.



Bad anti-aliasing

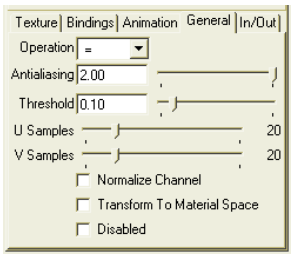


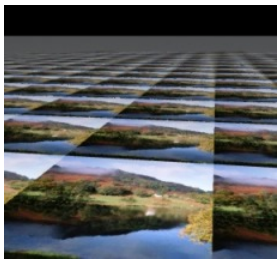
Adjusting anti-aliasing

In the material's property window check the Advanced option to see the internal structure of the material. The tree-like VSL diagram shows the structure of the material. You can see an item 'Color=Texture(Map coords)'. Select it by clicking it. The lower part of the property window now shows several tabs. Select the General tab.

Set the Anti-aliasing value to 1. It is a good default value to start with. The resolution of the logo image is some hundred pixels in both directions. When such a texture covers only few pixels in the final rendering many samples are needed to compute it. Optimally, the amount of samples should be the amount of details, i.e. the resolution of the image. Usually this would be very slow to compute. Setting the U samples and V samples to 10 tells the renderer to compute 100 (10*10) samples in the densest areas.

Render the view to test the result. If the quality is not good enough at the horizon, increase the anti-aliasing value by one. If the densest areas are well rendered, but there are problematic areas closer to the camera, halve the Threshold value from 1.0 to 0.5. The Threshold defines the minimal texture density where the texture anti-aliasing starts. If these adjustments are not enough to give a good quality everywhere, continue to increase the U and V samples.





Good anti-aliasing achieved

Material Classes

Realsoft 3D has an open architecture, which means that a material object can have any kind of property or interface. This chapter gives an overview of the material types included in the program. You can find the collection of installed material classes by opening the New sub menu from the popup menu of the select window's material section.

VSL Material

This is the most powerful and commonly used material type of the program. All materials, which you can find in the default material library, are of this type. The reason why VSL materials are so powerful is that the low level interface of the VSL materials is a programming language. Advanced users can therefore implement unusual and customized material effects. Don't be alarmed by the term 'programming language', you do not have to write complex formulas and algorithms, because VSL materials can be built using wizards and an easy drag and drop interface. Consequently, it is easy to get started. Nevertheless, in order to fully utilize the possibilities of VSL, some programming experience is required.

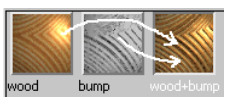
Using VSL materials is very easy. VSL makes it possible to build an optimal interface for each material type. You can see this by examining the materials of the default library in the property window.

You can find more information about this material type from the VSL specific section of this chapter.

Level Material

The Level material allows you to combine existing materials to construct more complex materials. Thanks to this, you can map several materials to an object using a single mapping object. The tutorial Blending Materials already showed one typical use of the level material: combining a material and a scope mask, which defines where the material has influence.

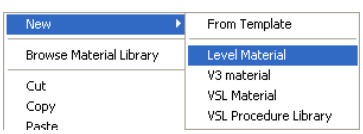
Creating level materials



Wood and bump are dragged and dropped into a level material. The result is a bumpy wood material

Level materials can be created from the select window's New menu (when the Materials tab is selected).

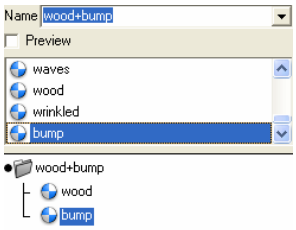
This material can refer to any number of other materials. The sub materials of a level material can be selected simply by dragging the desired materials into the level material in question.



Create a Level Material

Modifying level materials

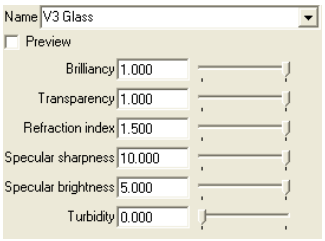
Level materials can be edited as usual: just double click the desired level material in the select window and the property window is opened allowing you to edit the material. Also the property window supports drag&drop. Drag the desired materials from the upper material list into the level material.



A level material consists of two sub materials: wood and bump

V3 Material

This material type implements the same kind of controls for optical material properties as Real3D. Brilliancy property defines the amount of mirror like reflection. The amount of diffuse shading is automatically computed from brilliancy: if brilliancy is zero, diffuse shading is at its maximum and vice versa. The material can be used for basic mirror, glass, etc. and it renders quickly.





Note

The more general VSL material allows all effects that V3 material can define, but V3 material is slightly faster to render.

3D Painting

This tutorial chapter introduces the 3D painting tools of Realsoft 3D.

Level: Beginner/Medium

The 3D painting system of Realsoft 3D allows you to paint the surface of any geometric object using various painting tools, such as freeform curves, rectangles and circles. The program includes apowerful material system for defining optical and other properties of objects.Furthermore, objects can be textured using various texture projection methods, such as volumetric mapping and UV mapping. In addition to standard mapping methods such as parallel and spherical mapping, the VSL language can be used to define customized and unusual mapping transformations. Realsoft 3D also allows any material property to be defined using any number of textures per surface. The system is very open and the possibilities are endless.

The 3D painting system of Realsoft 3D is designed to support all these features.

It allows you to paint any surface property (including user defined channels), such as color, transparency, refraction or UV coordinates. The painting target may have multiple textures, and regardless of the mapping method, they all are taken into account.

Creating a football

A well-known problem in texturing spherical objects is that it is very difficult to draw textures for the poles of the sphere due to the fact that the UV space in the poles is collapsed to a single point. 3D painting solves this problem.

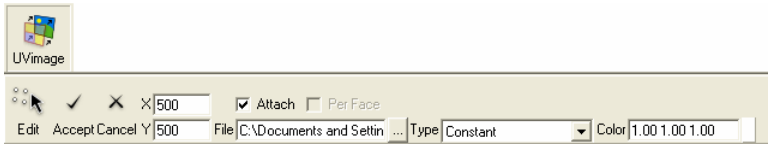
1. Create an analytic sphere using the Analytic/Sphere tool.

2. Activate the UVimage tool. You can find it from the Materials section of the toolbar.

3. Specify the desired texture resolution (for example 500*500) and set Type to Constant color. Set the Color field to white (1,1,1), which is a suitable base color for the football.

4. Define a suitable File name for the image, for example 'myfootball.bmp'. Click Accept.

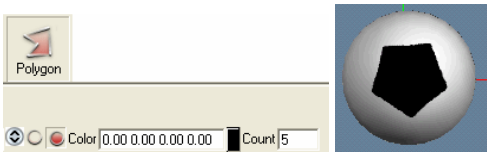
This creates and attaches a new blank white image to the selected objects (=sphere) using direct UV map. You can verify this at the sphere's Property window Col tab.



Activate the UVimage tool and initialize a texture for the football

5. Switch to the top view. If shaded OpenGL is not yet on, activate it from the view control bar. This way you will see in real time how the shaded surface gets painted.

6. Activate the Polygon painting tool. Select Filled polygon and set Count to five. The Color field should display black color, because we are going to paint the black areas of the football. Now, enter two points through the view window to specify center and radius for the polygon to be painted.



Activate the Polygon painting tool, turn Filled on and set Count to five. Draw the first polygon onto the north pole.

7. Select Camera/Front from the view's popup menu. Then select Again from the popup menu to draw the second polygon on a new side of the sphere. The Again menu activates the polygon paint tool with exactly the same options as before, so you can just draw the polygon.

Repeat step 7, each time selecting a different view direction from the Camera menu, until you have painted all sides of the sphere.



3D Painted sphere

8. Go to the Select window's Image tab. Double click the name of the image you created with the UVimage tool to view the generated texture.

You can save the generated image using the File/Save As function. If you save the entire project, the generated image (myfoolball.bmp) will be saved also. If you only want to save the painted image, set Sections to Selected and set only the Images check box.

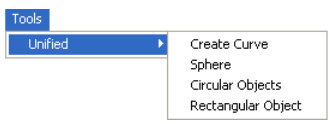


Image generated by 3D Painting

Using curve, rectangle and circle tools for 3D painting

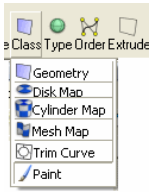
The 3D painting tool set, included in the Materials section of the toolbar, contains specialized tools for most of the common geometric shapes, like circles, rectangles, freeform curves, and polygon lines.

In addition to this, you can also use the four tools included in the Unified menu for 3D painting. Setting the Class option to Paint activates 3D painting mode. Instead of creating new objects, the tools paint the specified geometry to those underlying objects, which have bitmap textures.



These four basic creation tools support the 3D painting option

The advantage of this painting method is that modeling oriented tools have more options for defining the shape. For example, if you want to paint a circle arc by defining three points through which the circle arc should pass:

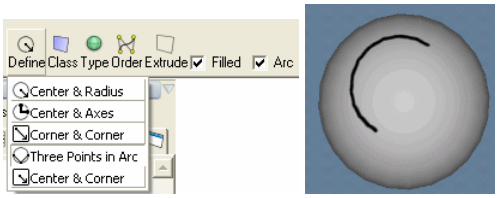


Set Class option to Paint

1. Create a sphere and attach a texture to it using the UVimage tool the same way as in the football example above.

2. Activate the Circular objects tool from the Unified menu. Set Class option to Paint.

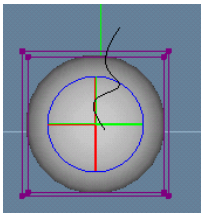
3. Set the Arc check box of the tool control bar. Set the Define method to Three Points in Arc and enter three points on the sphere through the view window.



Check the Arc option and set the Define method to Three Points in Arc. Painted arc shown on the right.

Using selected objects for painting

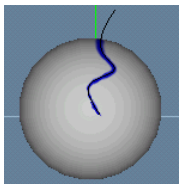
You can use any geometric object as a pen for 3D painting.



A sphere and a curve selected

1. Create an analytic sphere and use the UVimage tool to create a blank white texture for the sphere.

2. Use the Curve tool (Note: Curve tool of the Unified menu, or the NURBS/Curve tool, but not the 3D painting tool) to create a curve over the sphere.



Curve painted to sphere

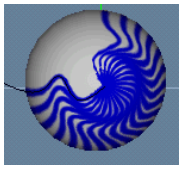


The Paint with Selected tool

3. Make sure the curve is selected. Go to the Materials section of the toolbar. Activate the paint with Selected tool and specify desired color and line width. If the curve is open like in the image above,select the unfilled drawing option. Then click Accept.

One advantage of using selected objects for painting is that you can use all geometry manipulation tools to control 3D painting.

For example, you can use the Macro system to paint a certain pattern over the object:



Repeat 'rotate + paint' macro to get the entire sphere painted

4. Activate Macro/Record from the main menu.

5. Rotate the pen curve a bit.

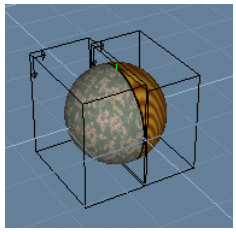
6. Apply the Selected paint tool.

7. Reset Macro/Record from the main menu and hit the 'Ctrl+g' keys a couple of times to execute the current macro.

Converting materials to direct UV mapped textures

The UVimage tool is very powerful and allows you to do much more than just creating new blank textures for objects. This example demonstrates how you can do 3D painting by mapping multiple materials to objects using standard mapping methods, such as parallel, cylinder and spherical maps.

Let's assume that you want to create an object with a marble left side and a wooden right side.



A sphere with two parallel maps: wood and marble.

1. Create an analytic sphere.

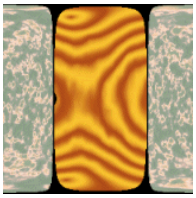
2. Go to the select window's material tab and map the two materials to the sphere by dropping the materials onto the view window. Move and scale the created parallel maps so that they split the sphere into two halves like in the image below.

3. Go to the object hierarchy tab of the select window. Select both parallel mapping objects, choose Properties from the popup menu. Go to the Spec tab of the property window and set all the Finite options. These options limit the mapped material inside the parallel map's cube like volume.



Select both mapping objects and set the Finite options through the Property Window/Spec tab

Rendering the view window shows the desired division between materials. However, the materials are procedural (defined by mathematical formulas). This means that if you want to export the scene, for example, to a polygon based game engine, the materials must be converted to bitmap images which then can be used as usual texture maps. Continue as follows:



Volumetric parallel mapped materials converted to a surface texture

4. Select the sphere and activate the UVimage tool. Set Type to Evaluate. Specify the desired resolution (e.g. 512*512) and file name, and click Accept.

This creates a new image with the specified name and attaches it to the sphere as a direct UV map. You may delete the two parallel mapping objects now.

The generated image can be loaded to any 2D painting program for further editing, etc.

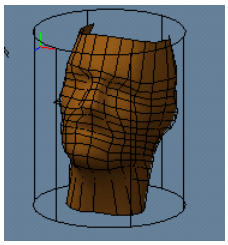
Painting SDS objects

A problem with subdivision objects is that they usually don't have a well defined default UV space which could be used to bind textures on them. For example, when modeling a head, the user starts from a basic shape like a cube or a sphere and uses extrude, subdivide and other SDS tools to add new faces and control points to the object. The tools do their best to guess sensible UV coordinates for the added points, but if the construction process is long and complicated, the UV space of the result object will not be well defined.

The following kind of problems may occur:

- UV coordinates of points may not be evenly distributed. A part of model that is big in 3D space may occupy only a little part of UV space. Consequently, it is not possible to paint small details to such parts of the model. Even the smallest point (one pixel in texture image) covers a wide area.
- Many parts of the model may have the same UV values. When painting to such an UV folded' part of the surface, other parts will get some paint as well.

Map2Obj tool allows you to redefine the UV space for objects. The general-purpose tool actually can define any surface property of SDS objects, but we will consider only UV coordinates in this tutorial.



The SDS face and a cylinder mapping object

The task is to paint a modeled SDS head. The first step is to make sure that the model has a decent UV space.

1. Model a head or load in 'models\sds\head'

2. Create a cylinder mapping object around the head from the top view (activate the Unified/Circular objects tool from the menu and set the Class option to Cylinder map). The material used is irrelevant.

3. Multi select the head and the cylinder mapping object (the order does not matter, because the tool recognizes mapping objects automatically) and activate the Map2Obj tool included in the Materials tab of the toolbar. Set the following options if not already set:

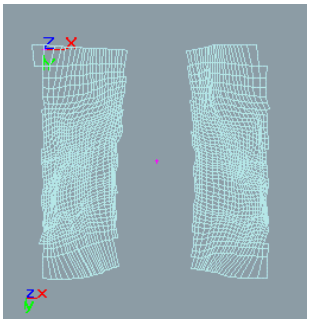
- Target: Points
- Source: Map Coords
- Destination: UV coords



Define pointwise UV coords

Then click Accept.

The above mentioned settings convert the material mapping space defined by the cylinder mapping object to pointwise UV coordinates of the SDS head. You can verify this through the UV Editor:



UV Space generated by the cylinder map

4. Select the head and open the Windows/UV Editor window from the main menu.

You can now fine-tune the UV coordinates. Set the head object to point edit mode from the tool control bar. Open the popup menu of the UV editor and make sure that Current UV is selected (it selects the pointwise UV editing instead of face specific coordinate editing). Also, make sure that the defined UV channel 'UV coords' is selected in the popup. SDS objects can namely have multiple UV channels per vertex. Then edit the control points in the UV editor.

The property window allows you to modify UV coordinates numerically:

5. Select the SDS control point(s) by clicking/dragging them through the View window of from the UV editor.

6. Go to the property window's Spec tab.

7. The UV field shows you the current uv value of the selected point(s). Enter a new value to this field.

A note for advanced users: Map2Obj can also be applied to selected points only. This option may be necessary in defining UV spaces for complex shapes. The target material of mapping object may also contain suitable Map coords transformations. For example, the simple transformation 'Map coords = Linear(Map coords)', 'multiply' = '0.5, 0.5, 0.5', 'add' = '0.5 0.5 0.5' in the Surface properties shader maps all the points to one quarter of the UV space. Similar transformations can also be defined using the transformation controls of mapping objects (included in the Property window's Spec tab).

Use your favorite painting program in 3D painting

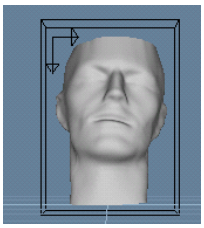
This is another example demonstrating the UV-image tool.

The exercise this time is adding a texture to a face. Painting a human face is not a trivial job, so you might want to use your favorite 2D painting program to do the actual painting.

1. Load in a model of a face (for example the included sample object 'models\SDS \head'). If the object is a subdivision object, make sure that UV coordinates are properly set (see the previous example).

2. Go to the Materials section of the select window. Create a new VSL material called 'face' using the New/From Template tool; the appropriate template is Image Maps/ Color Map. Leave the texture file name empty at this point.

3. Map the face material to the face model using parallel mapping as shown in the adjacent image.



A face and a parallel mapping object

4. Zoom in so that the parallel mapping and the face fill the view window. Set view's projection to Parallel and turn shaded OpenGL on.

5. Now, copy and paste the image in the view window to your favorite 2D painting program (such as PhotoShop) by holding down the Alt key and the Print Screen key on your keyboard. Create a new file in Photoshop (File size is automatically the size of the clipboard) and Paste in the image. You can also use Render/Save to File menu of the view's popup to save the contents of the view window and then load it into the paint program.

6. Paint over the shaded face image. When done, use the Crop tool to create a new image, which contains only the pixels inside the parallel mapping.

7. Save the image with a suitable name.

8. Go back to Realsoft 3D. Select the face material and open the property window. Set File name of the material to the image you painted and saved. Close the property window.

9. Ray trace the view window. The face image you painted in your favorite 2D painting program now matches exactly the face model. However, it is mapped using parallel mapping and cannot follow possible deformations of the face. Therefore we have to convert the parallel map to an UV map.

10. Activate the UVimage tool: Set Type to Evaluate. Specify a suitable resolution (can be the same as when you painted the image) and a file name and click Accept.

You have now converted the parallel map material to a direct UV texture. Delete the parallel map object and the face material. Another possible workflow for doing the actual painting in a 2D painting program is the following:

1. Use the UVimage tool to initialize a direct UV texture for the model the same way as in the first Football example.

2. Use the 3D painting tools (the Curve painting tool is quite suitable) to draw rough outlines of important areas like eyes, mouth, eyebrows, etc.

3. Save the project. Start the paint program you like most and load in the UV texture, which already has good drawing guides for the face. Finish the face drawing.

Paste colors on objects

Consider the following problem: an image of a face should be painted on another, completely different face. The image cannot be simply projected onto the face with a parallel mapping, because the size of the nose may be different, etc.

The Paste Colors on Objects tool provides a solution to this problem. This tool allows you to use any 3D object as a projector for projecting the image on the target surface. The image is projected using the current view projection: the target object will get the color of that point of the projector object that appears in the same location in the view window.

1. Load/Create a model of a face. If the face is a subdivision object, set up a proper UV space as instructed in the example 'Painting SDS objects' above. You may use one of the provided face objects, for example 'models/ head.r3d'. Turn the view window to show the face from the front.

2. Create an empty UV map for the face using the UVimage tool (see the Football example above). Name the texture file as 'matchedface'.

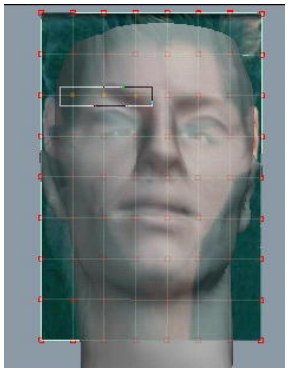
2. Create a rectangular Nurbs mesh of say, 10 times 10 points. The denser the mesh, the more accurately you can define the projection. Move the mesh behind the face, but in front of the back of the head.

3. Open the property window, go to the Col tab and enter the name of the face image to the Texture field.

4. Select the face object and using the property window's Wire tab, set the No Shading option for it. Alternatively, you may set the 'Forced visibility' wire setting of the projector mesh to, say 0.5. This forces the texture mapped projector surface to show through the shaded face. In both cases the purposes is to see the shape of the face object and the face image at the same time.

5. Go back to view the face from the front view and turn shaded OpenGL on. You will now see the face image through the wire frame of the face object.

6. Select the rectangular mesh. Then use deformation tools and point editing to move the points of the mesh to match the image to the face object.



Fitting a face texture to a model of another face by deforming the projector mesh

7. When done, make sure the nurbs mesh is selected. Go to Materials toolbar section and click the Paste Colors tool. Specify a desired sampling quality (e.g. 50*50) and accept the tool. The texture map on the nurbs mesh is projected onto the surface of the face.

You can now delete the projector mesh. You can find the matched face image in the Images tab of the Select window and view it using the property window.

Because the target of painting (or its UV space) can be very curved, the tool cannot guess a good sampling quality. Therefore, the tool asks the user to specify the number of samples to be taken. High sampling values may take a long time to compute. If the suggested quality 50*50 wasn't good enough, you can activate the Paste Colors tool again using higher Resolution values.

Note that you can paint any number of images to the head this way. Just rotate the view and repeat the steps 2-7 to paint ears, wrinkles, hair etc. over the head. New paste operations will overwrite earlier painting on overlapping areas.

Preparing an object for 3D painting

In the previous examples, we used the UVimage tool to create and attach a new texture to an object. You can also attach an existing image to an object and paint to it with the 3D painting tools.

There are two ways to connect the target image of 3D painting to the target object of painting:

- Direct UV map
- Material system

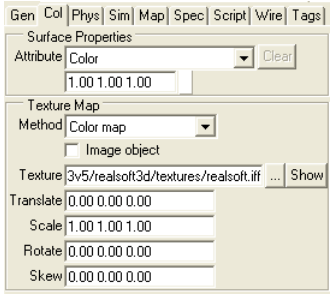
Direct UV Map

The UVimage tool used in the football example above assigned the created image to the sphere by using the direct uv map method. If you want to use an existing image, you can do this manually as follows:

1. Select the target object and open the Property window.

2. Go to the color tab and specify the desired image in the Texture field.

The specified image is now used as a direct UV mapped texture for the object and all 3D painting tools can be used to paint the object.



Use myimage.jpg as a direct uv mapped texture

Material System

Also bitmap textures included in materials and mapped to objects using the mapping tools (parallel, cylinder, spherical, etc.) can be painted with the 3D painting system. The advantage is that you can paint simultaneously to multiple images. Furthermore, the texture map objects can define not only color, but also any other information. You can paint transparency, bumps, and even visibility of materials to surfaces this way, as shown in the advanced tutorials.

For example, to paint bumps to a sphere (sample project: tutorprojects/material/3dpaint/paintbumps)

1. Select the sphere.

2. Activate the UVimage tool. Set Type to Constant color, Color to black and clear the Attach option. Resolution can be e.g. 256*256. Set an appropriate file name, for example 'textures\testbumps'. This creates an empty black texture map, but does not attach it to the sphere as a color map.

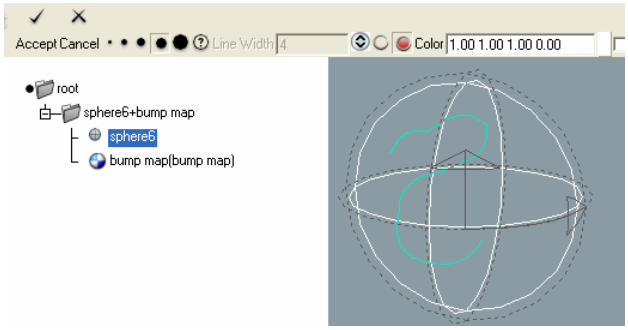


Initializing a bumpmap

3. Go to the Materials section of the select window and select New/From Template from the popup menu. When the file browser appears, select Image Maps/Bump Map and click Open. Open the Property window and set File name to 'textures\testbumps' or whichever name you used in the previous step. Close the Property window.

4. Drag and drop the newbump material to the view window. This creates a spherical mapping that assigns the bumpmap to the sphere.

5. Activate, for example, the Curve painting tool from the Materials section of the toolbar. Select a medium size brush, set non-filled line type and set the color to white. The brighter the color, the higher bumps will be drawn. Then press the left mouse button down over the sphere and draw a curve by dragging the mouse (LMB down). Release the mouse button when the curve shape is ready. Render the view window to verify the result. Use 'Again' from the view popup to draw new curves.





Drawing bumps

You can also initialize an empty black UV texture map, 3D paint to it and afterwards change the color map to a bumpmap. The advantage of this approach is that you can see the result of painting in real time using shaded OpenGL.



Note

When painting an object that has multiple textures using 3D painting tools, it may be necessary to protect all but one texture at a time from being modified. For example, an object may already have a color map, which must not change during bump painting. This can be done simply by selecting all mapping objects that you do not wish to paint to, and set Hit Invisible from the property window's Gen tab. If the object has a direct UV map (i.e. Texture field set in the object property window's Col tab) and you wish not to paint on it, the name string must be temporarily cleared. Cut the name, then past it back when painting is done. The trick works because the render engine that handles 3D painting is the same as the ray tracing engine.

Visual Shading Language

Introduction to Visual Shading Language

Welcome to *Visual Shading Language (VSL)*

VSL provides you with power equal to programming languages. It allows you to define very advanced material effects, such as procedural bump maps, anisotropic lighting effects or light radiation that makes the target object transparent!

Despite the fact that VSL is actually a programming language, you don't have to be a programmer in order to use it. Thanks to an easy-to-use drag&drop interface, most of the power of VSL is accessible to users with no programming background.

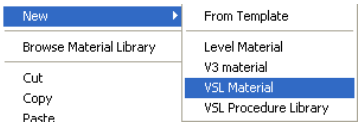
This tutorial shows you how to get started in creating fancy material effects with VSL.

Tutorial level: *Beginner*

Constant red

In this example we will create a material which turns the surface red.

1. First we need a new VSL material. To create one, switch to the Materials tab in the select window, open the pop-up menu by clicking the right mouse button, and pick New/VSL material from the list.

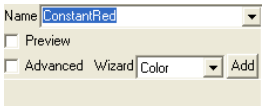


Create a new VSL material

A new black icon is created into the material tab with the default 'MaterialX' name. If Show Previews option is off (you can check it from the select window's popup menu), you will find the new material at the end of the material name list. Slowly double click the new material icon to activate the rename tool. Enter the new name 'ConstantRed' and press Enter.



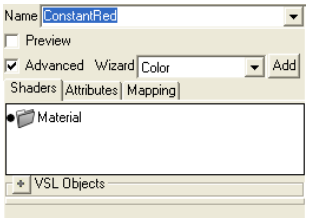
A new empty material is inserted into the material tab of the select window.



A blank material

2. Double click the new black material icon and the property window is opened. The created material does not define any material properties. Therefore, its property window is empty.

3. Check the Advanced check box in the top left edge of the material property window. This opens the actual VSL editor.



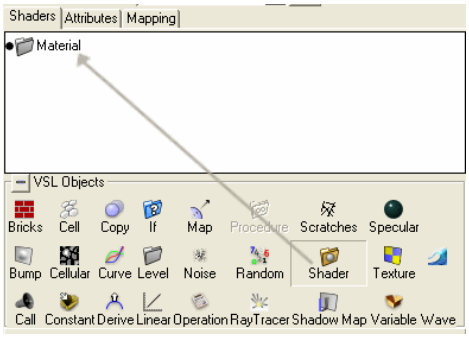
The VSL editor

4. Click the VSL Objects button (the small + sign). This shows you the VSL objects, which you can use to define material properties.



Available VSL objects

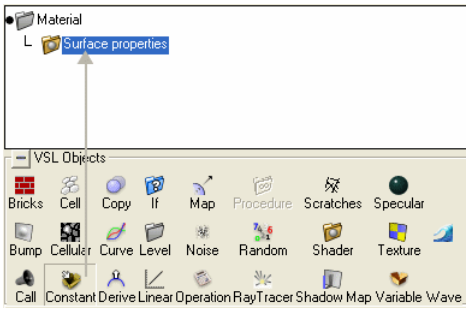
5. Our goal was to create a material, which simply turns surfaces red. This means that we want to define the *surface properties* of objects. Drag the Shader icon from the Available Objects and drop it over the 'Material' node appearing at the top part of the window. Note how the mouse pointer changes to indicate when dropping is allowed. This creates a new node to the VSL tree. The new node has the name 'Surface properties' and is automatically selected, as the highlighting of the name indicates.



Drag and drop the Shader icon into the Material node

Look at the lower part of the VSL editor. It always shows the properties of the selected VSL object. The current selected object, 'Shader', has only one property, 'Type', which is by default 'Surface properties'. Click the right side of the 'Type' gadget to open the whole list of available types; it includes names such as 'Light properties', 'Volume illumination', and so on.

Many of the shader types are easy to understand from the short but descriptive name. If you want to make an object red, you need a 'Surface properties' shader, because 'red color ' is a property of the surface (red paint, etc.). If you want to define how fog reacts in light, you need a 'Volume illumination' shader. Makes sense, doesn't it?



Drop a Constant object to Surface properties

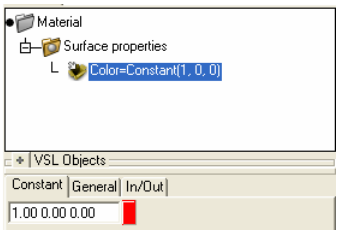
The default type 'Surface properties' is exactly what we need, so do not change it, just close the type list by clicking the right side of the gadget again. Surface properties is the most frequently used shader, and that is why it is the default type for all newly created shaders.

5. Drag & drop the Constant icon from the Available Objects box inside the Surface properties node. Again, a new node is inserted into the VSL tree.

6. The new Constant node is again selected, and the lower part of the window shows you its properties. The default color value is black. Click the small sample color square and change the color to red.

7. Check the Preview button to update the icon in the material select tab and to view the effect of the material.

Your first VSL material is now ready.



Surface color set to red

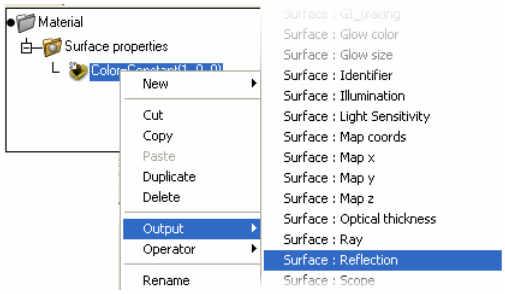
Red reflections

Next we modify the ConstantRed material slightly.

In the example above, we used a VSL object 'Color=Constant(1,0,0)' to achieve a red surface. To be exact, we defined such a property that makes an object appear red when ordinary white light illuminates it. This property is called the *diffuse color* of an object, and it has a very strong impact for the appearance of most real world objects and materials (cloth, paper, matte-like surfaces in general). Theoretically, diffuse color describes how much and in which color tiny irregularities and bumps of a surface scatter light around.

Diffuse color alone is not sufficient when we try to simulate real world materials and surfaces. Another well-known property, which describes what happens when light hits a surface, is the amount of mirror-like reflection. We will experiment with this next;

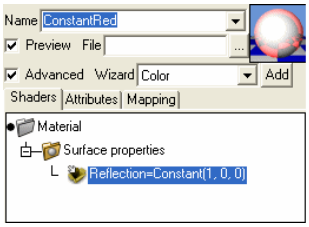
Using the previous material, make sure that the Constant VSL object in the material hierarchy is selected (its name is highlighted). Move the mouse pointer over the area, which shows the VSL structure of the material, and open the popup menu using the right mouse button. Open the Output sub menu. It includes a long list of items: Surface:Alpha, Surface:Color, and so on. Select Surface:Reflection from the list.



Selecting the output channel for a Constant VSL object

The long menu list shows the properties, which describe the type of the surface: the amount of diffuse color, reflection, transparency, etc. Such properties are called *Channels* in Realsoft 3D. You can control the appearance of a material by writing desired values to suitable channels, just like we did above (first diffuse color and now mirror reflection).

Some of the channels have names that may look hard to understand: UV coords, Map Z, for example. These channels contain information computed by the rendering engine and they are made available for the user so that he/she can find out what is going on during the rendering and ultimately, customize all the rendering actions.



Red reflections and white diffuse color

Let's stop the theoretical considerations and continue the tutorial. The Constant VSL object now defines the amount and color of mirror reflection. Consequently, the preview window shows the material again white, except that there may be some red reflections visible. All objects have white diffuse color by default. Thanks to this, we can see newly created objects in rendering without a need to define colors for them. Nothing in the VSL structure constructed so far changes this default setting, and therefore the diffuse color is white again.

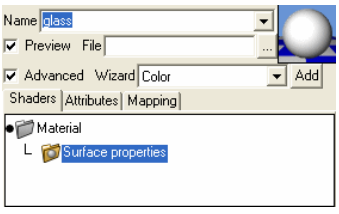
The reflections look strange, because the floor below is white and blue, but reflections show it red. Anyway, this is what we defined: the color value of the constant object is still red from the first VSL tutorial. As a matter of fact, although this is an unusual effect, even stranger phenomena occur in real world materials. Therefore, it is important that we have the freedom to implement the wildest imaginable shading effects using VSL.

Glass

Next we build a bit more complex material: glass. There is a glass like material available in the material library, but it is useful to learn how such a material is defined step by step.

Select New\VSL material from the select window's popup menu and rename the created blank material as Glass.

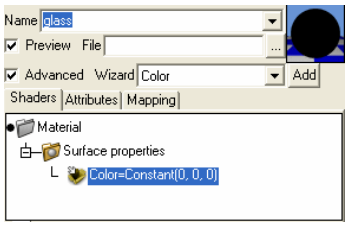
Open the property window unless already opened. The newly created material is automatically selected. The Advanced and Preview options should also be set from the previous example; if not, activate them.



The starting point

The first step is the same as in ConstantRed: drop a Shader from available VSL objects into the root level material node. Most glass properties can be defined in Surface properties, and therefore we do not have to change the default type of the shader.

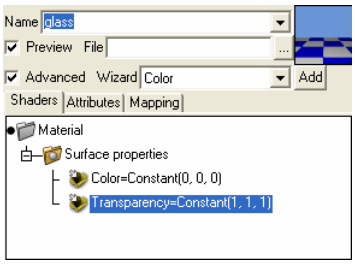
So far, the material has only the default properties: white diffuse color, thanks to which we can see the preview sphere. In the example image above, the simple preview sphere has a blue and white checker floor below it. This makes it easier to see how the material will interact with the environment. The preview project is fully customizable and can be changed (see the material preview tutorial for details).



Black diffuse color

Drop a Constant object from the available objects into Surface properties. The constant object is inserted, assigning the constant black value to the (diffuse) color channel. Consequently, the preview turns black. This is what we want, because the surface of glass is so smooth that it does not scatter light randomly around but lets it through and reflects it in certain exact angles. Keep in mind that random scattering is the source of matte-like diffuse shading.

Drop another Constant object into the Surface properties shader. Open the popup, and change the Output channel to Surface:Transparency. Changing the output channel was explained in the Red reflections example above. Nothing happens yet, because the constant transparency value is still black. Change it to white (1 1 1) using the gadget on the bottom of the VSL editor. The preview sphere disappears.



Fully transparent sphere

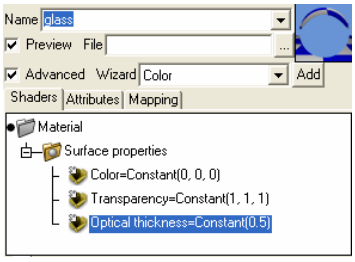
This kind of material is very boring, because applying it to an object is almost the same as deleting the object - nothing remains visible. The material lacks an important property of real glass, namely the ability to refract light. Let's fix this.

Drop a new constant object to surface properties and change the output to Surface:Optical thickness. The default thickness is zero - the thickness of empty space. Change the value of constant object to 0.5 using the gadget at the bottom of the property window. Note that optical thickness is defined by a single value, whereas color has three components: red, green and blue. The constant object automatically changes its value gadget to match the sub channel dimension of the output channel.

Note that you can compute the refraction factor from the optical thickness value using the simple formula

```
Refraction index = 1/(1-thickness)
```

Assigning the value 0.5 gives refraction index = 1/(1-0.5) = 2. The material mimics real glass in this respect. Note that as you become familiar with VSL, you can easily build a material, which uses the preferred way of measuring the refraction.

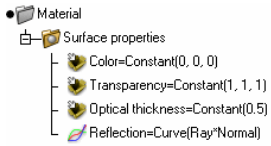


Refracting glass

The preview window changes, but you may not see any refraction yet. The preview window is actually a small view window and you can rotate, pan and zoom around just like in normal view windows. Move the mouse over the preview window, press Alt key down and drag with the right mouse button to rotate the view upwards. At certain angles, the refraction of the blue floor becomes visible. It is quite extreme, because the glass lens here is fully spherical. Think about eyeglasses with spherical lenses - certainly hard to wear and they turn the world upside down.

The material now looks like glass, but we can still improve it. You have probably noticed that all real world glass objects not only refract light but also reflect it. The mirror reflection is sometimes weak, but it exists. The previous example already showed how to make an object reflective. This time we will define the amount of reflection in an advanced manner: reflection should depend on the viewing angle towards the surface. The smaller the angle becomes, the more reflection appears (which happens in the real world as well).

Drop a Curve object from Available Objects into Surface properties. The preview turns funny because the default settings are all wrong.

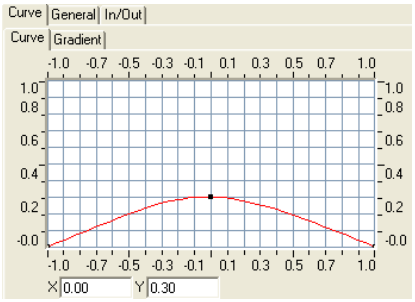


Reflection defined by viewing angle

Open the popup menu and change Output to Surface:Reflection. Open the popup again. There is one new entry, Input 0. The Curve VSL object computes values to the output channel by applying a user drawn curve function to values read from an input channel. Set Input 0 to Surface: Ray*Normal. This channel is maintained by the ray tracer and measures exactly the property we want: how perpendicular the ray from the camera is to the surface. The range is from -1 to 1, the middle value zero being the case when rays travel along the surface (perpendicular to the surface normal).

The curve gadget at the bottom of the property window shows a straight line ascending from zero to one. Move the mouse pointer over the curve and open the popup menu. Select Set Minimum and Maximum values from it. Change the MinX from zero to -1, hit Enter and close the window. Now the curve covers the whole range of possible Surface:Ray*Normal values.

Edit the curve shape so that the maximum height is in the middle (see the picture below). You can select existing points by clicking them or with a drag box the same way as when editing geometric objects. To add a new point, simply drag the curve from a point holding the left mouse button down. You can delete the unnecessary points using the Delete function from the popup menu. Note that there are actually three curves, but edit them all to a same shape, because we do not want color distortions this time - white should look white in reflections. The the start and end values of the curve can be slightly above zero, which makes the surface reflective everywhere.

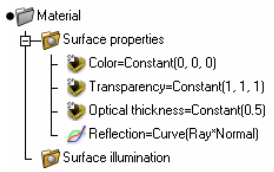


The curve that defines reflection

Now the reflection is ready, but you may not see much change in the simple preview scene in the small window. Nevertheless, this kind of subtle detail is important when building truly photorealistic scenes.

The final step is adding some shine to our glass.

Drop a Shader object to the Material node. Change the Type from Surface properties to Surface illumination.



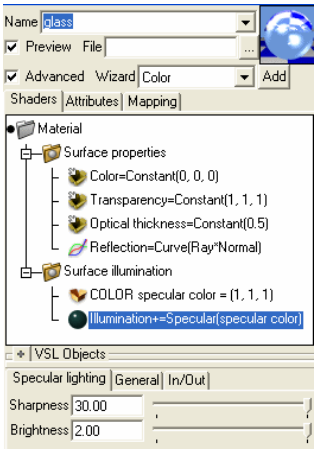
Surface illumination shader added

Drop a Specular object to the Surface illumination shader. Take a look at the inserted Specular object: It uses the Color channel as an input the same way as the curve object above uses Ray*Normal. Usually this works fine, but now the color is set to black (the first Constant assignment in surface properties) and therefore the default input does not work in this case.

Drop a Variable object on top of the Specular node in the VSL structure. Wait until the mouse pointer shows an up arrow indicating that the new object will be inserted before the specular object. If the Variable object was added after Specular, you can drag and drop it into a new position in the VSL hierarchy; just watch the mouse cursor when dropping to find out the correct end position. Actually variables can be inserted anywhere in the hierarchy and they are equally visible within the parent level, but declaring variables in the beginning of the level makes VSL code more readable.

Using the property controls of Variable, change its name to SpecularColor, set the Initialize checkbox and define a value 1,1,1 to the Initial Value gadget.

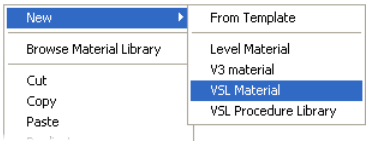
Click the Specular object in the VSL tree to activate it. Open the popup menu and change Input 0 from Color to SpecularColor. Now the specular object uses white color defined by the SpecularColor variable to compute the highlights. You can see this in the preview window. Adjust the sharpness and brightness values of the Specular object until the highlights are suitable. The glass material is now ready.



Glass with specular highlights

VSL Wizards

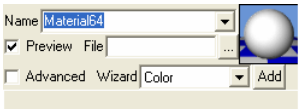
The easiest way to create new kinds of VSL materials is to use VSL wizards. These wizards are available in the VSL editor. Wizards construct the necessary material structure automatically, and insert suitable controls for easy use.



Create a new material from the select window's material tab.

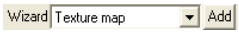
For example, to create a marble like material using wizards:

1. Switch to the material tab of the select window and create a new VSL material using the select window's popup menu.



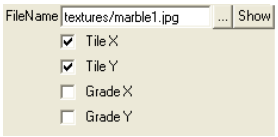
Property window with Preview enabled

2. Double click the created blank material icon to open the property window. Check the Preview option so that you can preview the material while the construction proceeds.



Add a texture map to the material

3. Select Texture map from the wizard list and press the Add button. New controls appear below the wizard list.

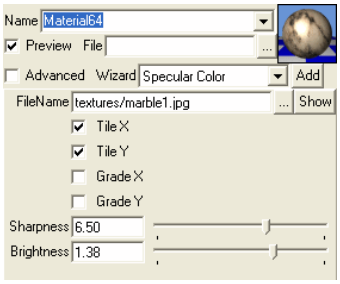


Texture map properties

4. Check the TileX and TileY options, and click browse to find a suitable texture map file. For example, textures/marble.

5. Select Specular Color from the wizard list, and press Add. New controls appear.

6. Adjust specular sharpness and brightness until the material looks suitable on the preview window.



A marble material created using the texture and the specular color wizards



Note

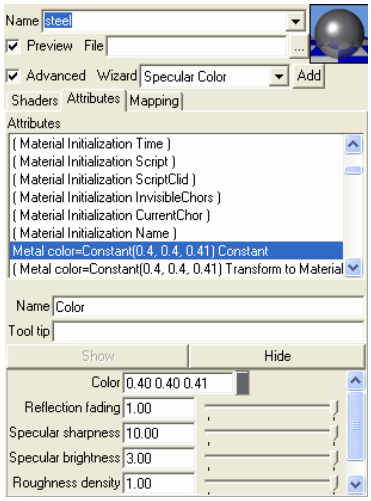
All wizards are editable - to change a color texture wizard is simply a case of changing the outputs; To change the texture wizard for example, to a fade map, change the output from Color to Fade. The wizards are supplied to make creating materials easier and as a base to work upon.

Building an Interface for a VSL Material

Consider the following situation: You have written a very powerful and nice material using VSL and you want to let other users to play with it. The VSL tree consists of 50 objects doing all kinds of fancy things. By changing certain attributes of certain VSL objects, you can control the appearance of the material.

So, how to expose only the key attributes (say irregularity and thickness of wood grains) to the end user and hide the unnecessary complexity from the user?

The VSL editor includes tools for solving this problem. It allows you to expose only selected attributes of a VSL material. Other users can adjust the properties of a VSL material without knowing anything about its internal structure.



Double click the desired property to show it in the easy-to-use interface

Setting up the interface

The Attributes tab in the VSL editor lists all the attributes of the VSL objects that has been inserted to the VSL tree of the currently selected material. The attribute names are the same or similar to the labels of gadgets, which are visible when you edit VSL objects. You can pick the suitable attributes and add them to the tailored interface.

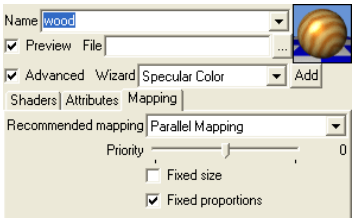
You can rename the exposed attributes using the Name field, in case the default label is not descriptive enough. For example, if you expose the same attribute of two different VSL objects, say the file name of a color texture map and the file name of a reflection map, renaming is necessary.

Selecting the preferred mapping method

The Mapping tab of the Advanced material properties contains settings, which define how the constructed material should be mapped to target objects. It is quite a common situation that a material should be always used together with a particular kind of mapping. A good example is wood: real world wood material consists of circles, which are parallel mapped along the axis of a tree. Therefore, only parallel mapping works well with wood.

The mapping specifications are used by the drag & drop system. The user can overrule them by selecting another mapping type from the mapping creation tools.

The following controls are available:



Selecting mapping options for drag & drop

- **Recommended Mapping**: select the preferred mapping type from this gadget.
- **Fixed size**: If set, the size of the mapping geometry does not automatically resize to match the size of the mapping target object.
- **Fixed proportions**: If set, the proportions (width, height and depth) of the mapping geometry do not automatically stretch to match the size of the target object. For example, parallel map always becomes a symmetric cube if this option is set.

If the recommended mapping is set to Any Mapping, drag & drop system creates a mapping that best matches the geometry of the target object. Spheres will get a spherical mapping, cylinders gets cylindrical mapping, etc.

Bump Mapping

Bump mapping is one of the key techniques when building realistic materials. It provides a quick and memory efficient way of adding fine detail to surfaces.

A traditional way to define a bump map is using a bitmap image to define the bumps: the brighter the color, the higher the bump. The major advantage of this approach is that the user can define the shape of bumps by drawing, which is intuitive. Disadvantages include high memory usage, finite resolution and the regular grid-like nature of the bumps.

Bumps can also be defined procedurally using VSL objects such as Noise or Wave. This method is suitable for representing various semi-regular real world surfaces, such as skin of a reptile, surface of a rock, waves

on water, etc. Procedural bumps can have very high accuracy (= level of detail), and they do not require much computer memory.

Realsoft 3D provides two different solutions for bump mapping:

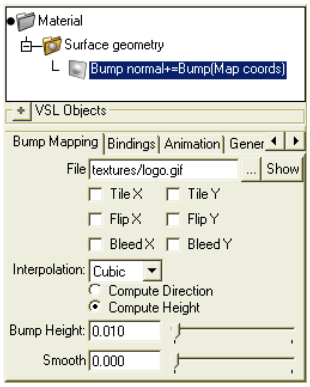
- Modifications of the Bump height channel in the Surface geometry shader
- Direct modifications of the Bump normal channel in the Surface properties shader

The first alternative is a general method and suitable for all kinds of bump mapping situations. Direct Bump normal modifications are more limited but are usually faster to compute. Both approaches are considered below in more detail.

Using the Bump height channel in a Surface geometry shader

This is the recommended bump mapping method because of its generality. The examples below show the structure of image based and procedural bump maps.

Image bump map



The Bump VSL object can be used for computing bump height values from a bitmap. The Texture object is not continuous enough for smooth bumps.

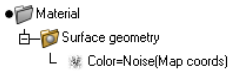
- Select Cubic interpolation for smooth bumps, or apply Smooth filter for even greater smoothing.
- The Compute height option should be set.
- Use the += operation to add new bumps to previous ones (to change this go to the General tab).
- Do not General/Normalize the bump height channel (see the General tab).

Example file: Materials/Image Maps/Bump Map

Procedural bump map

The image on the right shows a simple procedural bump map. The height of the bumps can be adjusted using the Amplitude control of the Noise object.





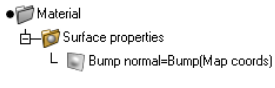
- Bump height units are measured in meters. Therefore, height values are typically very small, from 0.001 to 0.01 (1 mm - 1 cm).
- The computed Bump height value should be added to the previous value (+= operation). This ensures that several bump maps can be combined without problems.

Example: Default material library/Wrinkled

Using the Bump normal channel in a Surface properties shader

Image bump map

The VSL structure of a typical image bump map is shown in the image.

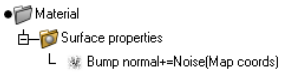


- The mapping geometry should match the shape of the target object: spherical mapping is suitable for a sphere, cylinder mapping for a cylinder, parallel mapping for a planar surface and default UV mapping for a freeform Nurbs surface. This method cannot be used for complex SDS shapes.

- General/Normalize option of the Bump operator should be set because the surface normal is needed in a unit form.
- General/Transform to material space option of the Bump object should be set so that the normal is modified in a mapping space relative way

Example file: tutorprojects/material/vsl/pixelbump

Procedural bump map



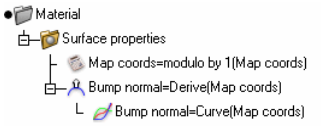
The VSL structure of a typical mathematical bump map is this:

- General/Normalize should be set.
- The noise should be centered on zero: set Noise/Base to -h and Noise/Amplitude to 2*h, where h is the magnitude of noise (for example 0.1).
- General/Transform to material space option is not necessary, because noise is similar in all directions. However, if the bump procedure defines regular shaped bumps, which should follow the target surface, the transform option should be set and the mapping geometry and surface geometry should match.
- The Noise operator works best with a parallel mapping, because the computed coordinates are continuous. Spherical mapping renders with a visible seam.

Example file: tutorprojects/material/vsl/mathbump

Curve object bumps

The Derive object can be used to transform an arbitrary VSL object to a bump map suitable form. For example, one can draw the shape of bumps using the curve object.



The first VSL object in Surface properties shader is Operation/Modulo by 1. This object computes repeating sequence of 0..1 coordinates and hence creates tiling of the bump pattern. The Curve object needs not have Normalize or Transform to mat space set, because the Derive object does this. Bump height can be adjusted by modifying the vertical scale of the curve.

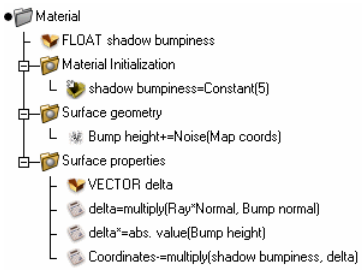
Bump map operates on two-dimensional surfaces, and therefore the third curve is unused. The first (red) curve controls bump height in the horizontal direction, the second (green) curve controls bumps vertically.

Objects such as Noise or Random do not require use of a Derive object, because Derive processed noise is just another noise field and the same applies to Random as well.

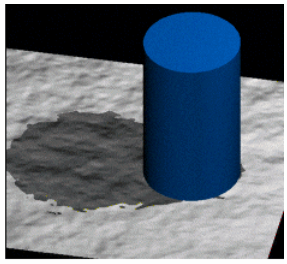
Example file: tutorprojects/material/vsl/curvebump

Bump map and shadows

An additional level of realism can be obtained by distorting the shadows on a bumped surface. The material presented here shows how to achieve this:



- The Material initialization shader is used to initialize the variable defining the amount of shadow distortion.
- Surface geometry computes the bump map in the usual manner.



A shadow on a bumpy surface

- In the Surface properties shader, the surface point is moved the amount defined by Bump height channel to the direction of the surface normal. This affects the illumination and shadow computations, which will take

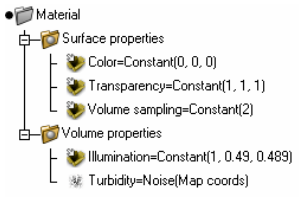
place later. Some extra care is taken to move the point to the camera side of the surface, not behind it. Moving the point behind the original surface would make the surface cast a shadow onto itself.

Example file: tutorprojects/material/vsl/bumpyshadows

Fog Examples

Plasma fog

Plasma like fog, that is, fog which is self illuminating, can be simulated by the following VSL structure:



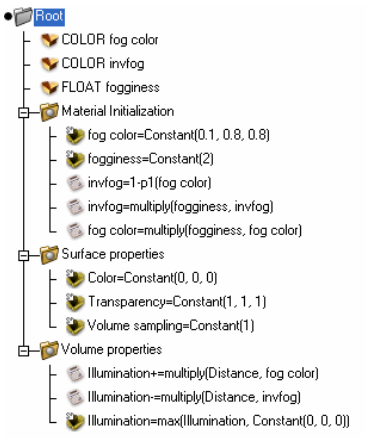
- The first two VSL objects in the Surface properties shader make the fog surface invisible. Assigning the value of 2 to the Volume sampling channel activates the volume effects: the interior of the fog cloud is studied by taking two samples at suitable intervals. For a simple fog like this, even a smaller value of 1 would suffice.
- In the Volume properties shader, a constant value is assigned to the illumination channel. The Illumination channel represents the total result of shading. This material assigns an illumination value directly. This means that light sources or shadows will not affect the fog, it appears self-illuminating.
- A Noise object defines the Turbidity property. Use a parallel mapping to map this fog material, then noise uses continuous coordinates without seams. Noise is not centered at zero because negative values would result in empty patches in the fog.
- Turbidity, the density of fog, is the strongest fog attribute in the sense that other illumination properties define the properties of one single fog particle, and the final result (color in the rendered image) is after all defined by turbidity. This makes sense: one tiny fog particle does not change the shading much, no matter how bright it is. There must be many such particles (=some amount of turbidity) before the contribution becomes noticeable.
- Remember to check Volumetric effects in the Render settings/Ray tracing options, to get the effect visible.

Example file: tutorprojects/material/vsl/plasma

Linear fog

The rendering system handles the blending of fog effects to other illumination automatically. The blending ratio is defined by the Turbidity property, which defines the density of the fog. The blending process is not linear: The longer the distance, which the light travels inside the fog, the stronger the influence of the fog becomes, but the fog never stops the ray completely.

The example image shows the structure of a material, which overrules this default fog shading. The fog effect defined by it is linear: if a layer of say 1 meter changes the illumination traveling through it by 50 %, then two meters of fog hides the background completely. The density of the fog defines the distance, which is required for full impenetrability.

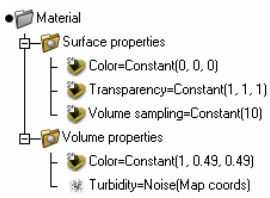


- Turbidity channel is not used at all. The material computes and assigns the fog effect to the illumination channel. Leaving turbidity to zero means that no default shading will take place.
- The fog is similar to the previous example in that it is self-illuminating.

Example file: tutorprojects/material/vsl/linearfog

Shadows in fog

The two fog types presented above are self-illuminating: light sources do not affect them at all. A fog that reacts with light can be obtained by the following VSL code:



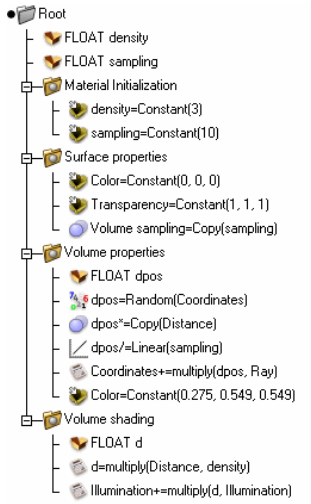
- The interaction with light sources is obtained by assigning a non-zero value to the Volume:Color channel in the Volume properties shader. The color channel defines the diffuse shading properties of the fog particles.
- The Volume sampling channel has now a much higher value than in the previous example. The higher the value, the more accurately the shape and details of the shadows inside the fog are rendered.
- Ensure that Volumetric effects and Lighting in volume are set in the Render settings/Ray tracing options.
- Only the light sources having the Volumetric option set (set by default) create volumetric effects.

Example file: tutorprojects/material/vsl/shadowfog

Additive fog

The Linear fog example above explained how to define custom fog shading for a self-illuminating plasma fog using the Volume properties shader. It is possible to customize the default fog shading of fog that reacts with light in the Volumetric shading shader, which is evaluated after fog illumination computations.

The material below shows one such example, additive fog. The fog is diffusely illuminated. It has the special property that it never darkens the illumination behind it. A normal fog does this: black smoke can hide flames of fire.



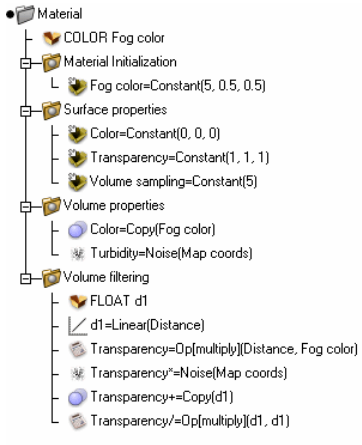
- Density is a parameter that the controls the thickness of the fog.
- Sampling defines how accurately the shadows in the fog are computed.
- The main parameters are initialized in the Material initialization shader.
- The Surface properties shader makes the fog surface transparent, removes diffuse shading from it (Color=black) so that the surface becomes clear, and launches volumetric sampling by copying the sampling parameter to the Volume sampling channel.
- The Volume properties shader does not set the Turbidity channel, which controls default shading of the fog. Instead, it moves the sample position randomly along the traced ray. The sampling density and distance control the amount of random movement. Therefore, the random variation keeps the samples inside the fog. The random factor improves quality and helps to detect small shadows.
- Volume shading computes fog illumination, which is proportional to density and distance. This illumination is simply added to any previous illumination. This makes the fog additive.

Example file: tutorprojects/material/vsl/additivefog

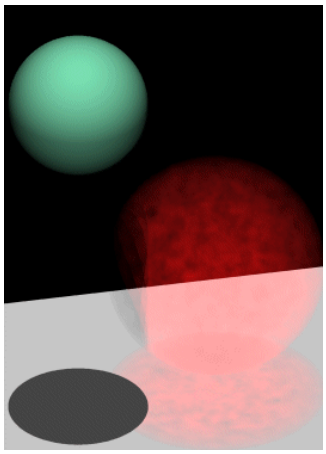
Shadows by fog

This example demonstrates how to use the Volume filtering shader. In this shader, it is possible to define how the interior of a material creates shadows. The shadowing property is defined by assigning suitable values to the VFilter:Transparency channel in the Volume filtering shader.

The VSL hierarchy is:



- The rather complex computations in the Volume filtering shader take care of several details: they produce a shadow, whose intensity is proportional to the square of the distance which the light travels in the fog. The intensity is modified by some Noise and the shadow color is smoothly faded to white at edges.
- The rather complex computations in the Volume filtering shader take care of several details: they produce a shadow, whose intensity is proportional to the square of the distance which the light travels in the fog. The intensity is modified by some Noise and the shadow color is smoothly faded to white at edges.



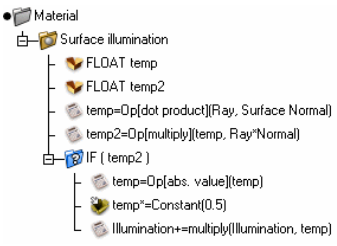
- The shadow in the example project is quite subtle and it is hard to tell if it is really the interior and not the surface, which creates the shadow. The difference becomes important in animations where objects and/or light sources are inside the fog. In other situations, it may be a better solution to create the shadow on the surface, because volumetric filtering is quite a time consuming feature. Especially, when both volumetric illumination and volumetric filtering are activated (like in the example), lots of computations are required, because the interior of the fog creates shadows in the interior of the fog. Therefore, the rendering time required by volumetric effects becomes proportional to the square of the sampling density - a 10 times higher sampling rate takes 100 times longer to render.
- Volumetric effects, Lighting in volume and Volume shadows in the Render settings/Ray tracing options must be enabled when rendering this example.

Example file: tutorprojects/material/vsl/shadowbyfog

Illumination Examples

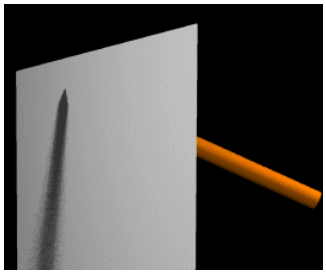
Paper

It is possible to define custom surface illumination effects by using a Surface illumination shader. The following material; paper, has the property that lighting (including shadows) is visible through the paper (to a certain extent).



The illumination shader computes the product of the dot products <ray*normal> and <lightray*normal>. This product is negative only when the surface is viewed from the opposite side of lighting. The If object notices this and computes illumination in those cases where default shading would leave surface without illumination.

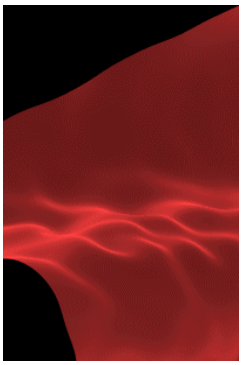
The alternate shading actions are very simple: the lighting intensity is modified by the lighting angle and then reduced by 50 %, because all the light cannot penetrate the paper.

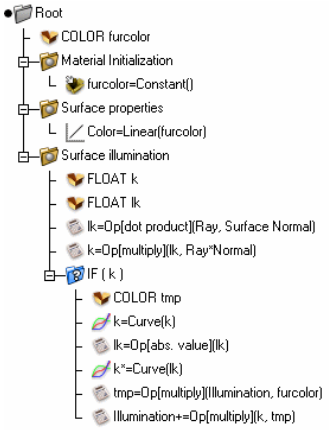


Example file: tutorprojects/material/vsl/paper

Fur

This example material simulates a surface covered by dense, short fur that shines in light.





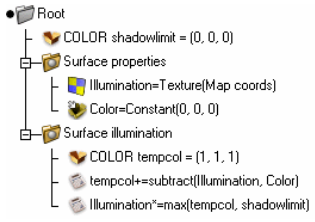
- The Surface properties shader includes only one object, which adds a low amount of diffuse color by scaling down the furcolor value defined in the Material Initialization shader.
- The actual trick is made in the Surface illumination shader: illumination is defined so that it reaches the maximum value when the small hairs are viewed from the side. This behavior is quite different from the usual diffuse shading.
- The If object tests, that light does not get through the surface.
- The two curve objects define the amount of illumination; by modifying these curves, surface appearance can be adjusted.

Example file: tutorprojects/material/vsl/shinyfur*Example file:* tutorprojects/material/vsl/silk

Matte shadows

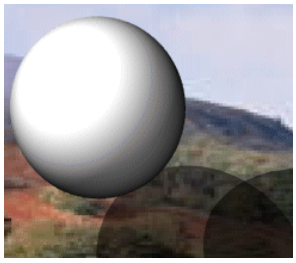
In the Surface illumination shader, both the original light properties of a light source and the amount of light reaching the examined surface point are available. It is possible to compare these two channels for creating advanced matte effects. In the example below, a photographed backdrop is not illuminated by the light sources (the photograph already includes illumination), but 3D objects still cast shadows on it.

1. Create the following VSL material:



2. Create a camera object using the camera tool.

3. Create a backdrop object using the backdrop tool. Use the material defined above.



4. Put some objects in front of the backdrop, and some light sources behind objects, so that objects cast shadows to the backdrop.

5. Take the camera view (View's popup Camera/Object View) and render.

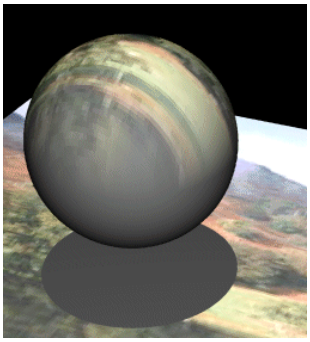
Some notes:

- In the Surface properties shader, the color of the backdrop is assigned directly to the illumination channel. The result will show the color in its unmodified state. Diffuse color is set to black to eliminate other shading effects.
- The Surface illumination shader first computes the amount of shadow. Negative values are truncated to zero, and the Surface:Illumination channel is scaled by the resulting factor.

- This example shows a situation where the backdrop image is in a digital form. We could also store the amount of shadow into the alpha channel. This would allow generation of shadows later to any background video sequence. It is also possible to map reflecting color to the areas where the backdrop image is supposed to reflect the environment and so on - the possibilities are endless.

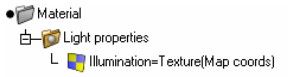
Example file: tutorprojects/material/vsl/matteshadows
Example file: tutorprojects/material/vsl/alphashadows

Slide projector



1. Create a light source.

2. Then apply the following material to it (for example using a parallel mapping):



Example file: tutorprojects/material/vsl/slideprojector



Note

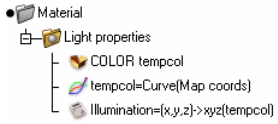
The mapped material defines light properties in the examined illuminated surface point, not on the surface of the mapping object. This means that the parallel projection used in the example above works best with a distant light source, because both project their influence in a similar way. A pyramid mapping matches with a point light source.

Custom light falloff rate

This example is recommended for those users who are familiar with polar space coordinates.

1. Create a point light source.

2. Then apply the following material to it using a spherical mapping. The origin point of the mapping should match the position of the light source.

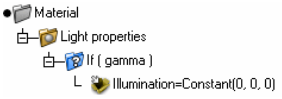


- The three curves of the curve shader control the light falloff with respect to polar coordinates. In other words, it is possible to draw the shape of the light beam created by the light source:
- The red curve controls the first polar parameter, heading. This may be constant in which case the light distribution is symmetric. By making peaks to it, light source may send more light up, to the left, etc.
- The green curve controls the pitching parameter. Zero pitching corresponds the direction of the spot axis, a pitching of 0.5 is 180 degrees from the axis. Curve values greater than 0.5 control directions behind the light source. For example: if the green curve has a high value in the beginning, the spotlight will have high intensity in the middle. By changing this parameter, it is possible to make the light source cast a light ring, for example.
- The blue curve is perhaps the most useful one: it defines the falloff by distance. Usually this curve decreases down to zero. The size of the sphere mapping defines the corresponding area in space. By resizing it, it is possible to adjust the area of influence of the light source.
- A more intuitive way to control the shape of a light beam is based on a slide projector idea. Draw two gray scale images, which show the shape and falloff of the light beam. Then map them to the light source from the side and from the top using a parallel mapping.

Example file: tutorprojects/material/vsl/curvelight

Gamma ray light sources

This example shows how to create a light source that casts special kind of light that illuminates only certain kind of objects.



If object tests if the gamma channel value is inside 0.01 and 1

1. Create a new channel of type Float using the New popup menu of the Channels tab of the select window. Rename it as gamma.

2. Create a light source and map the material of the upper image to it.

3. Create an object and map the material shown in the lower image to it (any mapping type is suitable):



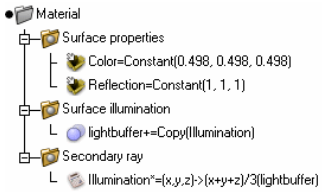
The Constant object assigns a value between 0.01 and 1 to the gamma channel

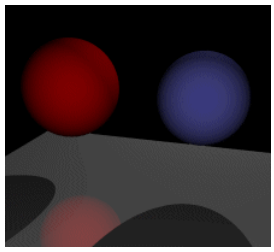
Only objects with gamma material become illuminated by gamma light.

Example file: tutorprojects/material/vsl/gammalight

Reflective in light

Light can define surface properties. For example, a light source can act as an X-ray radiator, which makes objects transparent. Or, like the example below shows, lighting can control the reflectivity of a surface.





Lightbuffer is a user defined color channel. The amount of light arriving from light sources is summed into it.

In the Secondary ray shader, the illumination of the traced reflection ray is scaled by the total sum of light.

Example file: tutorprojects/material/vsl/lightmakesreflective

Miscellaneous Examples

Clip mapping

Clip mapping is the effect where a material removes the underlying surface. The following example demonstrates this:

The If object test can be, for example, a test if the red signal is dark enough:




```
X inside (0, 0.5)
Y always
Z always
```

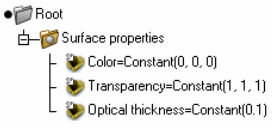
Setting the Distance channel to a negative value removes the surface. There is a VSL wizard available for clip mapping, as well as a template in the Image Maps folder.

Example file: tutorprojects/material/vsl/clipmap

Simple glass

This is the structure of a very simple glass-like material:

- Color is set to black. This removes diffuse illumination, which would otherwise practically hide the transparency of the material.
- Transparency is set to white (1,1,1). All light gets through the surface.
- Refraction is defined by optical thickness channel, 0.1 is suitable for glass.

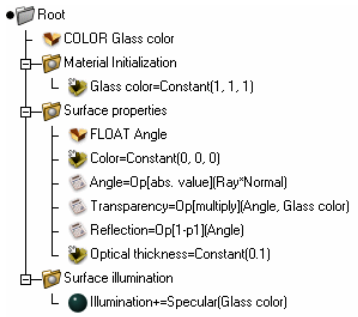


Example file: tutorprojects/material/vsl/simpleglass

Realistic glass

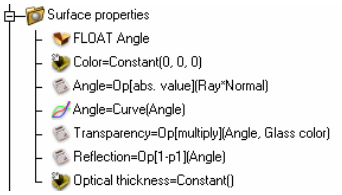
The glass material of the image is created by the Glass VSL wizard. It produces realistic looking glass.

- Glass color is defined as a constant, which is initialized in the Material initialization shader. Other needed colors are consistently computed from this variable.
- In the Surface properties shader, transparency is defined as a function of the angle in which the ray hits the surface. Perpendicular rays penetrate the surface well, whereas rays that have the direction of the surface are reflected. The channel ray*normal measures surface-ray angle: the channel value is +1 or -1 for orthogonal rays and approaches zero as the angle gets smaller.



Possible improvements:

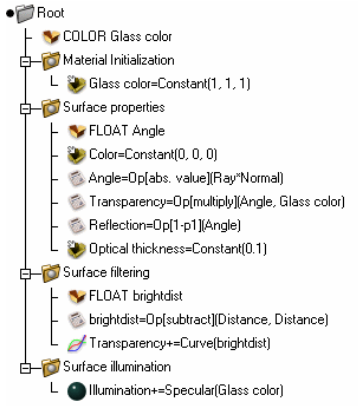
If accurate reflectivity control is required, a curve object can be added to surface properties. By adjusting the curve, surface reflectivity can be easily modified as a function of angle.



The Surface filtering shader can be used to control the glass shadow creation. The adjacent image shows an example. Here the difference

```
Op[subtract](light:distance, filter:distance)
```

measures the distance behind the glass. The curve object then adds some brightness according to this distance, so that there is actually a focal distance where the effect appears brightest. Finally, if the 3 curves that modify transparency are not identical, a colorful spectrum shadow is created!

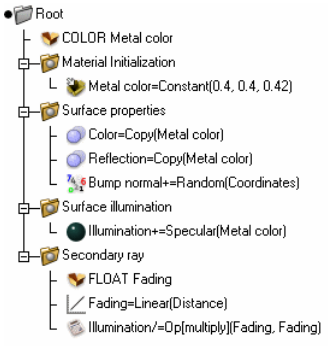


Example file: tutorprojects/material/vsl/spectralglass

Steel

In this example, the VSL structure of a metal created by the steel wizard is examined;

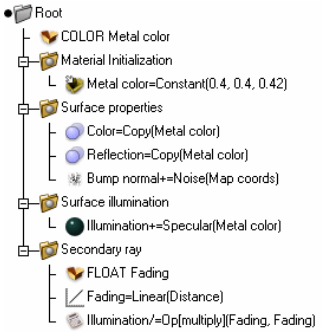
- The Random object that modifies the bump normal channel makes the surface appear rough. General/Normalize option should be set. The Random object's distribution should be zero-centered: base = -0.5*amplitude.



- Reflections on the metal surface are faded by distance using a Secondary ray shader. First, the distance of the reflection is modified by a simple linear transformation, then this value is squared to obtain faster, non-linear falloff rate for fading, and finally the color of reflection (found in traced ray:illumination) is divided by the value.

Example file: tutorprojects/material/vsl/steel

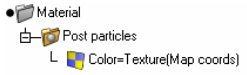
By replacing the Random object with a Noise object, the rough appearance of the previous material can be changed to a brushed metal appearance. The brush effect requires that the mapping used defines one dimension that is strongly compressed. The cylinder mapping axis is made very short for this purpose in the example project.



Example file: tutorprojects/material/vsl/brushedsteel

Texture mapped post processing particles

The color of particles can be defined using a VSL material:



1. Create a set of 1D particles.
2. Map the particle disks post effect to the particles with a default map.
3. Map the material shown on the right to them with a parallel map

A Post particles shader defines properties for particles that are rendered using a post effect such as Lens flare or Particle disks. Properties for Scan line type particles are defined using a Surface properties shader.

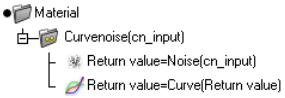
Example file: tutorprojects/material/vsl/texturedparticles

User defined VSL procedures

The program includes tools for creating new user defined VSL objects. Frequently needed algorithms can be collected into a VSL procedure library. The procedures can be called from VSL materials. For example:

1. Create a new procedure library by using the menu New/VSL Procedure Library of the material tab of the select window. Open the Properties window. The created procedure library contents are displayed.
2. Drop a Procedure object to the Root node of the VSL tree.
3. Select the created Procedure() item. Rename it as Curvenoise using the Name field on the bottom of the VSL window.
4. Change to the Parameters tab and press Add. A new parameter appears to the previously empty parameter list. Rename the parameter, for example, as cn_input using the Parameter name field.
5. Drop a Noise object to the Curvenoise procedure. Change the input parameter of Noise to cn_input. Output should be 'Return value'.

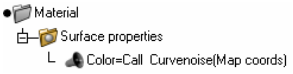
6. Drop a Curve object to the Curvenoise object. Both output and input parameter should be 'Return value'. Modify the curve a bit. The procedure is now ready. Its structure is shown in the image.



7. Activate the select window and create a new VSL material. Set Advanced option to see its contents on the property window.

8. Drop a shader to the VSL tree of the new material. The default, Surface properties, is suitable.

   Drop a Call object to the surface properties shader and select it. Pick Curvenoise from the displayed list of available procedures. The default input and output parameters are suitable. They can be changed using the VSL editor's popup menu.

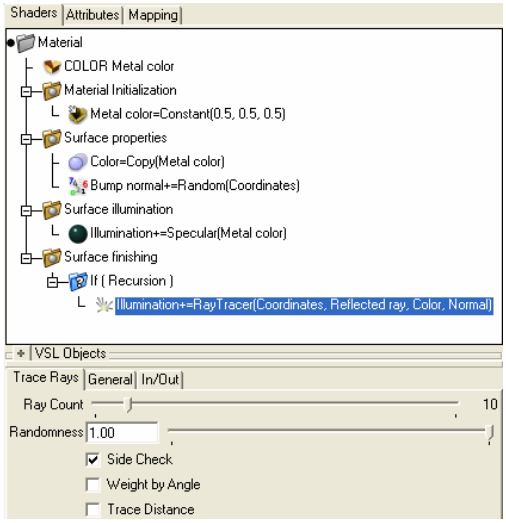


9. Map the material to a test object using a parallel mapping and render.

Example file: tutorprojects/material/vsl/vslprocedure

Aluminium with blurred reflections

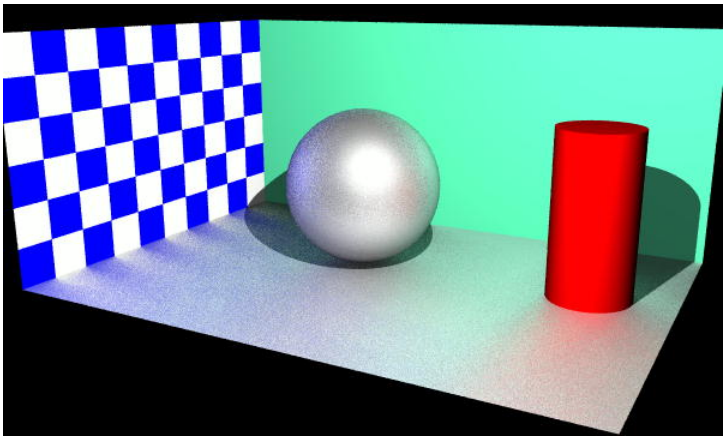
Customized ray tracing effects can be generated using the Raytrace VSL object. The aluminum like material, whose tructure is shown in the image, generates strongly blurred reflections.



Blurring is created by a Raytrace object in a Surface finishing shader. The Raytrace object traces 5 random rays around the reflection direction. Each sample direction is strongly randomized. The rendered image shows how reflections remain recognizeable only at a very close range from the aluminium surface.

The image quality depends heavily on the amount of traced samples. It is easy to adjust the amount of traced rays, but high values should be used with care. If the scene contains aluminium surfaces that reflect each other, computation task (and hence rendering time) increases very rapidly when sampling rate grows. Already at a recursion depth 3 and sampling rate 10 one single aluminium surface evaluation can generate 10*10*10 = 1000 ray trace operations!

Both recursion depth and recursion threshold should then be adjusted to keep the rendering time tolerable. The material itself can also define a local recursion limit: the Raytracer object can be placed inside a If level which tests that recursion depth is less than 1.01.



It is also possible to define a post processing configuration which filters away the noise of a shading component before adding it to the image. For details, see the example about post processed illumination in the Rendering / Advanced Illumination Examples chapter of the manual.

Example file: tutorprojects/material/vsl/blurredaluminium

Type Casting

There are three kinds of channels and variables available: color, float and vector. Most of the VSL objects repeat the same calculation to all sub channels independently. The user does not have to worry about the data type differences when building VSL code. VSL automatically changes data types to a similar form so that the computation can be performed.

For example, in a VSL expression 'color = wave(coordinates)' both input and output channel are 3 dimensional. Therefore, the computation goes as:

color.red = sin(coordinates.x)

color.green = sin(coordinates.y)

color.blue = sin(coordinates.z)

The expression 'color = wave(distance)' (distance is one dimensional value) is computed as:

color.red = sin(distance)

color.green = sin(distance)

color.blue = sin(distance)

The expression 'alpha = wave(distance)' (both alpha and distance are one dimensional) is simply:

alpha = sin(distance)

and 'alpha = wave(coordinates)' goes as:

alpha = sin(coordinates.x)

Similarly, if the object, by its nature, always outputs a single floating point value, the result is copied to all output sub channels. So, 'ray = length(normal)' computes:

ray.x = length(normal)

ray.y = length(normal)

ray.z = length(normal)

Parameters

The number of input parameters of VSL objects varies. 'Constant' has none, whereas 'Linear interpolation' of 'Operation' object has three. The parameters can be set either by selecting the object and using the popup menu, or from the in/out tab of the VSL editor.

An object can have only one output parameter. Nevertheless, VSL objects can also change values of their input parameters. Because of automatic type conversions, the original address of a parameter is not always passed in a VSL object call. Instead, the system copies the (possibly modified) converted value back after the execution to allow multiple output channels.

This principle is important to know when passing a parameter several times to a shader that also modifiers several parameters. The final value is not defined by the order the object uses them internally, but by the order they are given in the parameter list. For example:

color = SpecialShaderThatChanges4Channels(color, color, color);

The actual output channel value assignment is made last and is most significant. After that, the assignment of the last (3rd) input parameter has second highest significance.

This should not be a problem because the order would be unknown anyway without explicit instructions; developers of new VSL objects should assume and follow the above defined convention.

Assigning Multiple Materials to SDS Objects

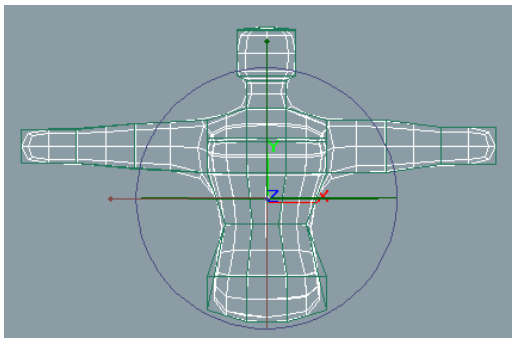
This example explains how to map materials to a subdivision object. Mapping is defined so that material patterns follow surface deformations (bending of skeletally controlled human figure, for example). You will also learn how to restrict each material to affect only desired parts of the surface.

Level: Medium/Advanced

Sample files: 'tutorprojects/material/sdsmaterial/torso' and 'tutorprojects/material/sdsmaterial/sdsmat'

First, model a simple human figure, like the upper body shape of the example image, which was created by extruding the head and two arms out from a 4*4*4 subdivision cube. Rename the figure as 'Torso'.

We will assign two materials, wood and brushed steel (both include color and bump patterns) to the torso. Therefore, we have to attach suitable mapping coordinates to the object (the patterns will follow the assigned coordinates). Both the wood and steel material are designed to work best with parallel mapped coordinates. Therefore, make sure that the torso is selected, go to the materials tab of the select window, drag and drop the wood material of the default material library into the view window. This creates a suitable parallel mapping. Ray trace the view and you will see a wooden figure.

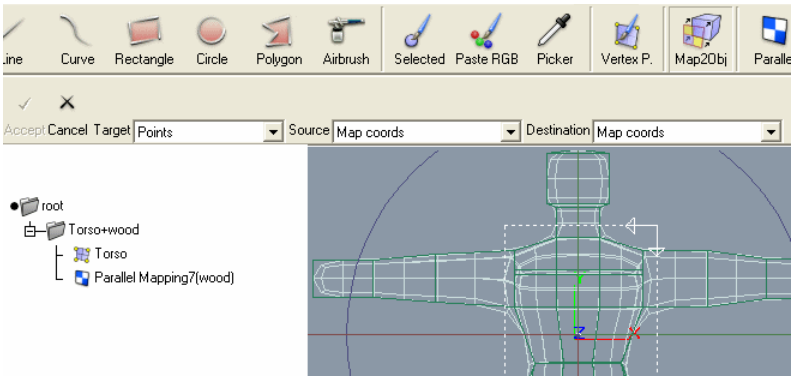


Subdivision modeled torso

Go back to the geometric objects tab of the select window. Open the hierarchy of the created torso+wood level. You can now adjust the density of the wood pattern by selecting the parallel mapping object and by applying the usual Transformation/Scale tool. The smaller the mapping object, the denser the wooden pattern.

This kind of wood mapping works fine for rigid objects. However, the strictly geometric parallel mapping cannot follow arbitrary flexible deformations. For example, if the arm of the torso were now bent, the arm would move across wood patterns, which looks strange in animations. Therefore, we have to convert geometrically defined parallel coordinates to pointwise defined parallel coordinates.

To do this, select the torso object and the parallel mapping from the select window. Go to the Materials tab of the toolbar and click the Map2Obj tool icon. The tool controls display various options: Target=points and Source=Map coords should already be selected by default, but change Destination to Map coords. Then click the Accept button.



Map2Obj tool stores mapping coordinates to surface points

You may now delete the parallel mapping object from the scene. Then select the parent level of the torso. Click the Default mapping tool icon on the toolbar (it is included in the same Materials section of the toolbar as the Map2Obj tool explained above). Make sure that the tool controls display Wood as the selected material and change Mapping type to Null. Then click Accept.

It you render the scene, the wooden torso should look pretty much the same as before. However, now the mapping coordinates follow nicely with all kinds of deformations. We used Null mapping above, because the surface itself computes mapping coordinates from the stored pointwise information and the default mapping should not overwrite this by some other coordinates.

Defining the influence area of a material using 3D painting

The next task is to limit the area of the wooden surface to some sub area of the torso. For example, let's try to write the text 'Wood' to the chest of the torso with wooden letters, the rest being the usual default material.

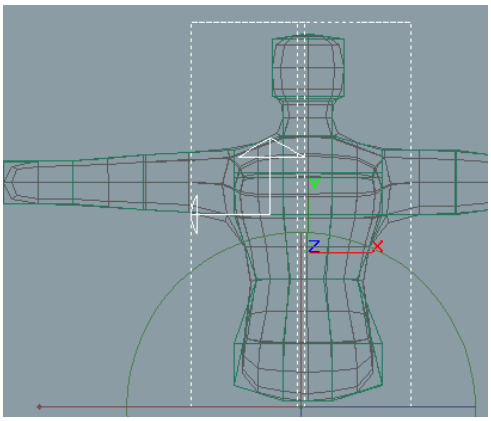
For this purpose, we need another set of pointwise coordinates. A nice way to define the text area is to use 3D painting. This method requires two-dimensional coordinates, which run along the surface. We can define these coordinates using the Map2Obj tool with a cylinder mapping.

Therefore, select the parent level of the torso, take a top view and pick Cylinder mapping from the toolbar's Materials section. Click in the middle of the torso, and shape the cylinder around the torso. The diameter is not important, but the seam direction sometimes is, so click a second time behind the torso to finish the cylinder mapping. The used material is not important, because this is just a temporary step.

Take a front view. Select the created cylinder mapping, extend and move it so that it covers the torso in the height direction (use Transformation/Move and Transformation/Extend tools or object handles to do this).

Then select the torso and the cylinder mapping and activate the Map2Obj tool. Target=Points and Source=Map coords are again suitable, but set Destination to UV coords. Then click the Accept button. Now the torso has parallel mapped pointwise mapping coordinates and cylinder mapped pointwise UV coordinates. Delete the cylinder map object.

The cylinder map does not always unwrap the surface of a target mesh properly. For example, if the arms of the torso were bent, some surface areas might get badly folded over other areas in unwrapped UV space. If this happens, the UV editor can be used to fix problematic areas.

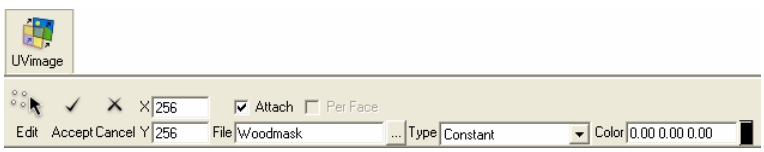


Cylinder mapping defines uv coordinates for surface points

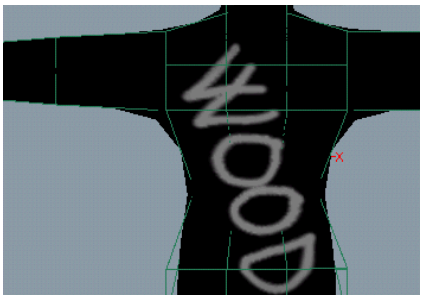
You can find the UV editor from the Windows menu. By comparing point selection on a normal view window and on the UV editor, it is easy to figure out how the cylinder map unwrapped the surface. Note that points can be selected on the normal view and moved on the UV editor.

We will assign materials to the middle chest of the torso, which is a neatly unwrapped area, so UV point editing is not necessary this time. If the purpose was to paint the arms, then some editing would be necessary, because the arms cover a very small area of the UV space and 3D painting is therefore inaccurate there. In a worst case, only a couple of pixels of the used texture map might cover the arms.

Next we define the wooden area. Select the torso and activate the UVimage tool (it is located in the Materials tab of the toolbar). This tool is for creating and assigning a bitmap texture to an object. The activated tool displays several new controls. The X and Y fields display the resolution of the image to be created. Default x=128, y=128 is a somewhat low resolution, so change it to x=256, y=256. Type=Constant is suitable; this means that the image will be initialized to a given color. Set the constant color to black by typing 0 0 0 to the Color field or by using the color selector which opens by clicking the small sample color square. The next option, Attach should also be set, so that we will see the image wrapped on the object as a color texture. Finally, enter a suitable File name, for example 'woodmask'. Then press the Accept button.



Turn shaded OpenGL on if not yet selected (you can find this option for example, from the view compass). The created texture map makes the object appear totally black. Activate Airbrush tool from the toolbar's Materials tab. Pick a medium size brush from the airbrush controls (for example 4 units wide brush) and set pure white color 1,1,1,1 into the color field. Then take a suitable look at the chest of the torso and draw the letters 'Wood' over the body. The white text becomes visible on the chest. Then press Accept to finish using the tool.



3D painted mask image

Now the mask image for the wooden area is ready and we will convert it from a color texture map to another form. Make sure that the torso is the selected object and select Object properties from the view window's popup menu. Select Col tab from the property window and empty the Texture field displaying the name 'woodmask.bmp'. Close the property window.

Building the materials

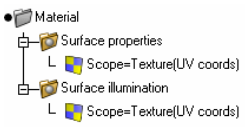
The next step is to create some materials. We could use easy to use templates for this purpose, but let's practice using the Visual Shading language a bit and build the materials step by step.

Go to the select window, switch to the material tab and pick New/VSL material from the select window's popup menu. A blank material appears in the material library. Double click the icon of the new material to open the property window. When the property window appears, first change the name appearing at the top to 'woodmask'. Enable the Advanced option and also open the Available objects frame, in case it was closed.

The wood material defines both surface properties and some illumination properties. Therefore we need to define the same properties for the mask material controlling the strength of wood. Drag and drop the Shader object twice onto the top most Material level node of the VSL hierarchy. Then activate the lower shader and change the Type field from Surface properties to Surface illumination.

Drag&drop a Texture object from available objects onto the Surface properties node. Open the popup menu on top of the created texture node and set Output to Surface:Scope and Input0 to Surface:UV coords. The scope channel is a special channel that is used to control the effect of other channels. The selected input channel UV channel, is the same point assigned channel that we used to bind the wood pattern to the torso. Now we use the same channel to bind the mask pattern to the model. Define the texture name woodmask.bmp to the File gadget at the lowest part of the property window (use the browse button labeled as '...' if you are not sure about the file name or path). Then enable the Tile X and Tile Y options. The mask image is wrapped only once over the torso (one single tile), but the UV coordinates must usually run continuously over the seams of a cylindrical surface like the torso. Tiling improves the continuity.

Also activate the Gradient X and Gradient Y options. The airbrush tool already created smooth edges to the painted area, but the gradient option improves the quality even more, so that the wooden area becomes smoothly blended to the background.



The material defining the influence of wood

Finally, select the texture object, then select Copy from the popup menu. Select the shader node Surface illumination and select Paste from the popup. The mask material is now ready and looks like in the adjacent image.

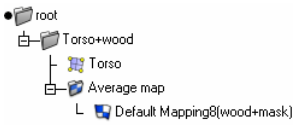
Go to the select window and select New/Level material from the material tab's popup menu. Rename the new material as 'Wood+mask' using the property window. The property window displays the available materials on the upper area and the contents of the level material on the lower area. Drag and drop the Wood and Woodmask materials from the upper material list onto the 'Wood+mask' node on the lower area. The result is shown in the exampel image.



A level material consisting of two sub materials

Average mapping

The next step is mapping the 'wood+mask' material to the torso. Change the select window to show the object hierarchy. Select the Wood default map object. Select property window's Spec tab. The top most gadget, Material, displays the target material of the mapping. Change it from Wood to Wood +mask level material. Then activate the Average map tool from the right end of the toolbar's Materials section and click Accept. This creates a new hierarchy level containing the default map object.



The hierarchy of the average map

The new Average map level is actually a special mapping object that averages the effect of materials mapped by its child mapping objects. Each material has a weight defined by the scope channel. If only one material is applied, an averaging object is not necessary, but we will also add some steel areas and the average map object makes sure that steel and wood are treated similarly. The following sample equation shows the difference between using and not using the average map, in a point where both wood and steel have scope 0.5:

Average map: result = 0.5*wood + 0.5*steel

Direct mapping: result = 0.5*(0.5*wood+0.5*default) + 0.5*steel

Without Average map, the blending of a new material to the previous situation is repeated until all materials have been evaluated and therefore the first material will have a weaker effect than the last material.

Close the property window and save the project with a suitable name. This step is important because project saving stores the changes in the images, including airbrush painting to the woodmask.bmp. Render the view window. You should see wooden text on a white torso.



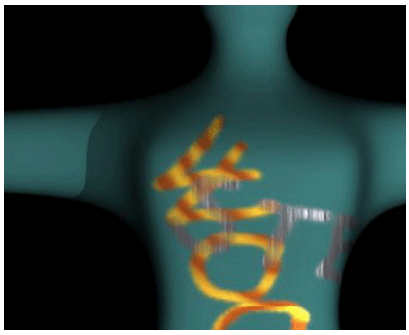
Masked wood area

Inserting more materials

Adding some steel areas happens exactly the same way. Before painting the steel areas, we have to protect the woodmask.bmp image temporarily, because 3D painting tools affect all bitmap textures hit by the brush. Therefore, select the Average map level object and open the object property window. Enable Invisible in photorealistic rendering option from the General tab. This makes the object in question inactive in all rendering actions, including 3D painting.

Next select the torso and use the UVimage tool to initialize a black bitmap called steelmask.bmp exactly the same way as woodmask.bmp above. Use white airbrush to paint the word 'Steel' to the torso. Note that the steel and wood texts may overlap without problems. When the steel mask is drawn and ready, go to property window's Col tab to remove the Texture field contents. The 'steelmask.bmp' was temporarily used as a color texture to see the painting on the surface, but from now on it will be used for masking the steel area. Use the color gadget to change the color of the torso to for example, darkish green, so that the mapped materials will be more clearly visible.

The easiest way to attach the masked steel material is to copy the already created wood items. Therefore, duplicate the material woodmask, rename the copy as steelmask and use the property window to examine its Advanced VSL hierarchy. Change the texture file name of the two texture VSL objects from woodmask.bmp to steelmask.bmp.



Blended materials

Create a new level material called 'steel+mask' and add the Brushsteel and steelmask materials into it (the same way as wood and woodmask above). Go to the object hierarchy tab of the select window and disable Invisible in photorealistic rendering property from the Average map level, so that the mapping starts working again in rendering. Make sure that the Average map level is selected and create a default mapping of type Null (see the beginning of this tutorial) referring to the new steel+mask material.

The tutorial project is now ready. Save it and render.

Modeling a Pot

Modeling the shape

Level: Beginner/Medium



The Finished Pot

This tutorial starts with creating a simple revolve, then texturing the result into a realistic pot. We make use of the Scope feature to show how to clip an image onto a surface that is already textured.

Start with a new project and select the front view, turn off perspective and reset the view.



Front View

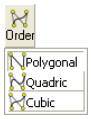


Parallel View



Reset View

In the NURBS tab, click the Curve icon. In the control bar select Polygonal curve by clicking the Order icon, and then select Polygonal from the drop down list. Leave all the options at their default settings - we want to draw an axis vertically that we can sweep our pot around. Start the curve at the lower middle of the view window; click once, then make another mouse click straight up the view and towards the top so the line is roughly vertical. (Holding the SHIFT key will keep your line vertical.)

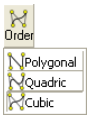


If you keep clicking at this point the curve can be made more complicated - but all we need is an axis so we will accept the curve at this point by pressing the Accept button in the toolbar (you can also right click and select accept from the pop-up menu). This finishes the creation of the curve.

Accept button

To simplify matters - it is always a good idea to name your objects. Under the select window, we can see that we have created a nurbs curve. By slowly double clicking the name next to the nurbs icon we can rename the object. Call it 'Axis'.





We now want to create the "Profile" of the pot. Think of this as a section through half of the pot that is 'swept' around our axis. Firstly though, we must select our axis - make sure it is highlighted before continuing. Under our NURBS tab again, click the Curve tool and select Cubic this time from the Order drop down list. (Polygonal curves have sharp edges - quadric and cubic curves are very smooth.)

We are going to use the axis curve we created as a point to rotate about when we make our profile. To do this, we need to select "Rotate" under our toolbar - click the Pen option to see the different methods - select Rotate.

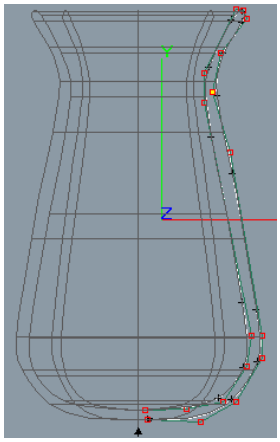


Starting at right hand side of our axis, click points from the base upwards to create a profile curve similar to the one shown. Each red point represents a mouse click. To get sharp edges in a nurbs curve, we need to put the points closer together. Hence the top has three closely grouped points. Note too that the curve draws both the outside, then goes inside the pot as well.

Do not worry too much about how accurately you place the points - you can edit them later. Notice too how you can see the creation of the revolve.

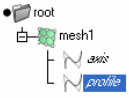
When you have finished creating all the points, click the Accept button again to finish creating the profile.

- ✓ *Accept*



The shape of your pot profile

Notice now that our level structure has changed and we now have a mesh in the view. We created the mesh by rotating our profile curve about the axis. The two curves are now put 'under' the mesh. Press the "+" icon next to the mesh to see the curves below. By editing these curves now will change the shape of our resultant mesh. Rename our nurbs curve to 'Profile'. Note that a mesh is a 3D surface - so when we render the image this is what we will see. Generally, curves by themselves are not rendered.



Rename the mesh to 'Pot'. Let's tweak the profile curve a bit. Select the profile curve, under the context sensitive toolbar a series of options are shown that we can do to this particular curve. Click the Edit button and we will move some points around to create the shape we want.



Edit

If you move the mouse cursor over any single point, the cursor changes to a cross. This means that you can now click on the point and move it around. Do this for each point until it looks roughly like the above image and has an even space between the inside and outside. You can now pan around the view and examine the pot you have created.



To get a better feel for the shape - turn on Open GL view to see the mesh better.





You can also raytrace the view by pressing teh render button in the view control bar.

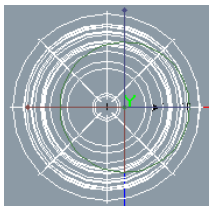
This is a good point to save your work. Under the File pull down menu, select Save As. Select the appropriate directory by clicking the '...' button next to the name field and Save the file as Pot_Part1.

Texturing our pot

Texturing a model is about applying materials to an object to achieve a given look and feel. These materials can be as simple as colors, or as complex as applying bumps, reflectivity, color and shine.

We can very simply drag and drop the standard glass material onto our pot to create a glass pot. Likewise, we can make it gold, plastic or chrome by using many of the standard materials.

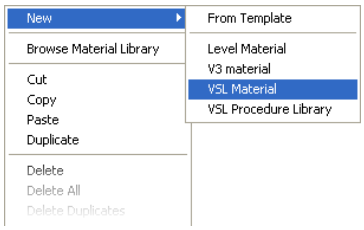
But we want the pot to use an image. This image could be a drawing, a color or a photograph. We will map this image onto our pot using the Cylindrical mapping method. We have many methods to apply the image to a model - parallel being the most obvious - it projects the image onto the model in the size and aspect you set when applying the material. Cylindrical mapping works on the same principle - but it wraps the image around the cylinder walls - much like the word COKE is written around a can. We can wrap 1 image right around the object, or we can make it Tile around. If I wanted COKE to appear twice, I would make it tile 2 times around.



In this example we are going to map a simple strip of color around our pot to make it look real. To do this we need to make this Strip tile about the pot 10 times. Here's how we do it:

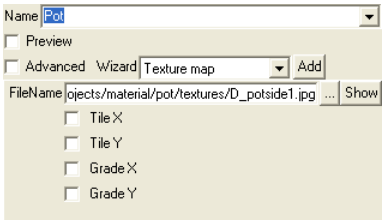
Load Pot_Part1 tutorial if its not already active (it is located at: tutorprojects/material/pot/Pot_Part1.r3d). Select the TOP view. Drag the Pot object mesh from the Select window and into the view window. This maximizes the object in the view window. (you can do this for any object including a camera.)

We should now have a wireframe view of the object like in the example image at the right.

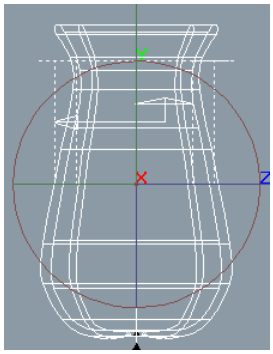


Let's go to the Materials tab: In the select window, select the Materials tab. Let's purge any unnecessary materials from this project - under the pull down menus, select Materials/Purge material Library. This deletes any UNUSED materials. If a material is being used, it is kept. This should delete all the materials as we have not assigned any yet.

Right Clicking in the blank area inside the material window should bring up our pop up menu. Select New/VSL material. This creates a new blank material for us to edit.



Select the new material and rename it (by slowly double clicking) to 'Pot'. Now quickly double click the material. This should bring up a new window called Property Window. This shows us all the selected objects properties. In this case, it shows us the properties for our material Pot which happens to be empty. Let's fix this by using one of the pull down wizards. Select Texture Map from the wizard pulldown, Add this by pressing the ADD button next to the wizard. This inserts a file requester. Let's grab our texture map image by clicking the '...' button. (Next to the SHOW button) Navigate to our tutorprojects/material/pot/textures directory, and select D_potside1.jpg.

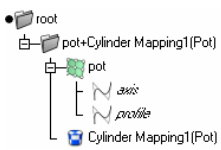


Let's map this material to our pot. In our material window, right click on the Pot material and select Map/Cylinder. This activates a tool to cylindrically map our texture to our model. It automatically assigns our material to whatever object is selected in the select window. Which should be our Pot Mesh as it is the only object.

To create our cylindrical mapping, first click at the center of the mesh, then just outside of the mesh. The result cylinder should just encompass our pot at its widest point. Rotate the view and see the result. Click the Side view. It should look similar to this image.

Note that the cylinder mapping is shown dotted. It is also smaller than our pot. We need the cylinder mapping to be just a bit taller than our pot. Let's size the mapping to fit the pot:

Go to the select window and examine what has happened to our structure. Our mesh has been dropped to a new level. Click the '+' icon next to the level to maximize its contents.

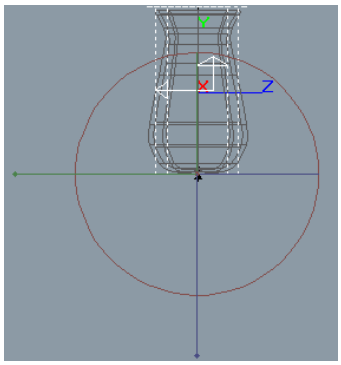


Note how the mapping is level vertically with the mesh. This mapping is now applied to anything inside this level - but nothing outside of it. Select the Cylinder Mapping(Pot). This selects just the mapping object. Let's use the handles of the object to position it correctly.

Put the mouse cursor over the blue line of the object handle - the cursor should change to a Move icon. Click the mouse on the blue line and move the object to just below our pot. This puts our object handle out of view. If needed, Pan the view so you can see the blue dot at the end of the object handles.



Now put the cursor on this blue dot at the bottom of our object handle - it should change to a pencil icon - this means we can scale the object about this axis. Click and drag down to scale our mapping object larger vertically. Keep modifying the object until the mapping is just larger than the pot. Note how the scaling or rotating of an object occurs about the handle, hence it scaled upward. If it was positioned in the center of the object, it would be scaled in both up and down directions. Our mapping should look like in the adjacent image.



Render the view to see the result. As you can see - we have applied our strip of color around the pot but it looks awfully dark! Let's add some lights next.



Go to Light Sources under the main toolbar to add some Point lightsources: Select Point - and click on the view for the light's center point, then another click to define how far the light should go.

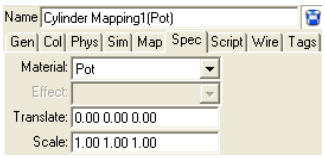
Point light tool

Add 2 or 3 lights around the pot. Render the view to see your progress and move the lights if needed. (Check your top and side views for light positions - move the lights by moving the mouse cursor over them and dragging).



Above is the image that we have stretched around the pot. (We have applied it vertically) It is very narrow and this results in a very blocky looking render of the pot. What we can do is tile this image around the pot a number of times. We change this by editing the Scale parameter of our mapping object;

Select the Cylinder mapping object, and click the Spec tab in our Property Window. (Remember we brought this window up by quickly double clicking any object). There are three numbers in the Scale field: '1 1 1'. This means that the mapping is mapped once in each three directions (X Y Z). If we want our strip to be repeated 10 times around, we actually need to make it 1/10th of the times in the X axis. Which means we need to punch in '0.1 1 1' There needs to be a space between each value as each value represents the X Y Z value. Try rendering the pot now and see what happens.



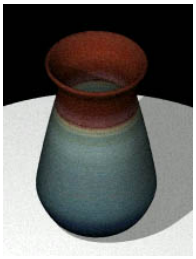


Our pot has only a strip of the image applied now! This is not what we want - it should be tiling around the object. We set this parameter in the material itself. Go back to our material tab and select out Pot material. In our Property window, there is a checkbox for Tile X. Turn this on and render again. Much better! We have now tiled the image around our pot 10 times.

Just for the sake of making our scene a little more interesting, Let's add a ground plane. In this case, let's put a disk underneath the pot so we can see some shadows. Under our Analytic tab, select the Disk icon. In a Top view, click the center point first (Center of the pot) then another point further out, well away from the pot. In a side view, move the disk to just below the pot.

Save your work as Pot_Part2.

Scope mapping



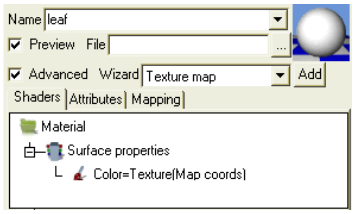
We have created a very basic pot with a simple cylindrical texture applied. We will continue fine tuning our material here and also map on a 'leaf' to the pot that shows some of the power of VSL.

Load "tutorprojects/material/pot/Pot_Part2.r3d". This scene has some lights and a ground applied to show off our pot better.

Let's continue to tweak our material and give the pot a few more realistic attributes.

Go to the Materials tab in the select window and select our Pot material. Double click the material if the Property window is not already up.

Let's use another wizard and add some 'Ping' to our pot, using the Specular Color wizard - Add it too.



Try rendering the view to see the difference. It renders as if it is made from gold. We want a more subtle appearance - so change the specular values to read:

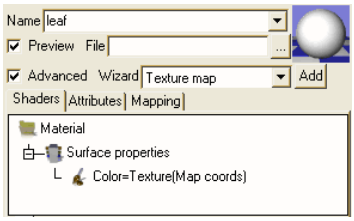
- Sharpness = 3.00
- Brightness = 0.80

We could go on to add bump and a bit of reflection if we wanted - but let's get started on a new material to map a leaf onto the pot.

Create a new material - right click in select material window and select New/VSL material.

Rename the material 'Leaf'. (We can do this by changing the name in the Property window, too.)

We have two modes when creating materials - Basic and Advanced. We are currently looking at the Basic mode in our property window. If we tick the Advanced button at the top left we can see exactly whats going on in this material. Let's do this.



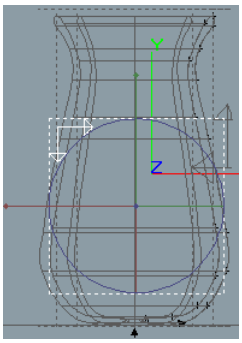
Use another Wizard and create a Texture Map. This creates a new surface property that is assigned to a texture. This texture is, by default, assigned to the color channel of an object. But we want it to be assigned to the Scope attribute in this case. The scope channel controls how much of a material is shown and how much is not. This is a powerful function that allows us to put any image onto an already mapped object (or unmapped) and clip out any undesirable background so we have only the desired portion shown on the image.

This has applications in assigning text over a material and removing the background so we only see the text. Assigning a complex window bitmap over a brick mapped wall, mapping a logo to a product, etc.

We cut our image out to do this - generally with an alpha channel (Extra information in an image that stores transparency data - typically Targa files are saved with an Alpha channel). We could also use a specific color value to do the same effect - but we have a Targa file with an alpha channel already setup.

Select our new texture map icon that has been placed in the properties window (red paintbrush icon). At the bottom of our properties window we see a file requester window similar to the basic version we used previously. Select the texture 'tutorprojects/material/pot/textures/D_potleaf+A2.tga' by pressing the '...' button.

If we were to map this texture onto our pot now, it would overwrite the material we created previously, so we need to setup our scope value so only where the leaf texture is shall be mapped onto our pot. Right click on the color=texture icon and select Copy, then Paste. This creates a duplicate of the texture we just assigned. Select the first texture map. Right click and under the Output menu select surface:scope. This changes the output from the texture map to go to the scope instead of the color.



And that is the basic structure of our scope material. Let's assign this to our pot and see what all this complicated VSL does. Go back to the select window. Click the front view and make our pot full screen (drag the mesh to the screen) Make sure that the pot level itself is selected and not the mesh as the material mapping will be put inside this level. Go to the material tab again and select our Leaf material.



Right click on the material and select Map/Parallel. Draw a rectangle roughly square - starting click at the top left and 2nd click bottom right. The map object is now inserted under our level as shown in the example image at the right. We can select the mapping object and scale/reposition it as necessary.

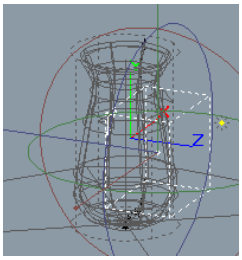
Rendering the view should yield a result similar to the adjacent image.

Make a few test renders around the object. Notice that we can move the leaf map around the object and it covers whatever is under it. But there is one small problem. The leaf appears on both sides of the pot! It is by default, mapped straight through and will continue going on forever. If we set the tile X + Y buttons on, it would repeat over the object as well. In this case, we only want 1 copy of the leaf, so we will set some attributes under the mapping object:

Select parallel mapping(leaf). Under the Spec tab in the properties window are three toggles - set them all ON - Finite X,Y,Z.



This limits the extent of our leaf material to within the boundaries of our mapping object. This is a very handy feature! Modify our parallel mapping object so it sits within one side of our pot and looks similar to the image at the right.

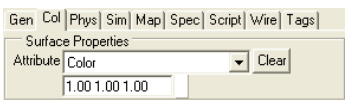


Finally, let's add a bump map to our material to give it a realistic raised brushstroke feel.

Go to the material tab in the select window. Select the leaf material and open the properties window. Using a wizard, add a bump map. In the name field, select the 'B_potleaf+A2.jpg'. Under the Bump Height field, enter '-1'.

Render your pot. Notice that all these materials have very subtle effects. To see what any of the parameters do - increase the values. (e.g. change the bump height to -5.)

To finish off our scene - let's change some of the values of the ground object without assigning a material to it. Go to the select window and select the Ground object. Under the Col tab of the properties window are various surface properties that we can change without the need to assign a material to. (A material would override these settings.)





Right click on the white square. Only one option appears: Expand. Click this. This brings up our color wheel. Select any color you like. Next to Attribute, Select the Reflection attribute. This uses a greyscale value to say how reflective an object is. Black is none, white is mirror. Select a dark grey.

Render the scene. The final result is shown in the example image.

Clip Mapping

Level: Intermediate

Clip mapping is the term used to describe using photographs or drawings as maps onto objects with an alpha channel. This creates the illusion of a detailed 3D object when in computer terms, all we have in the scene is one simple polygon.



On the left is our original image, on the right is the alpha channel that we use to "cut out" the horse.

The advantage is obvious - having a realistic looking render with a minimum of complexity in our scene. Clip mapping is particularly useful with objects such as people, trees, cars or animals. Any object that would have a high polygon count and does not need to be animated could become a good clip map instead.

There are three primary methods to assign a clip map in Realsoft 3D -

1. Use an alpha channel from a Targa file - this allows one to get various levels of subtle transparency. Targas have the advantage of being able to store the alpha channel in the same image file (32 bit = transparency channel, 24 bit = color only.)

2. Use a certain color as a clip channel. The solid BLACK background would be transparent in the tree show in the example image on the right side.



3. Use two separate images - one for color and one for the alpha channel.

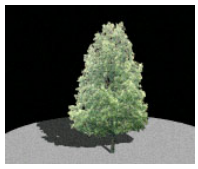
Color clipped Tree

Let's start with the most basic method first - color clip maps.

1. Create a new scene.

2. Create an analytical rectangle - any size.

3. Create a new material, use a template. Select Clipmap from the Image maps directory. Select the image "Tree1.jpg" from the tutorprojects\material\clipmap\textures directory. Map the material on the rectangle using a parallel mapping



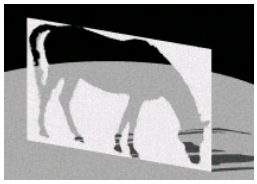
4. Render the image. At this point you can see that the default color range has used the black color to grey to define the transparency. This is probably too much, so let's select a darker color grey (just above black) for the Clip to field. Note that we have no color information on our object yet either - we have only the clip component.

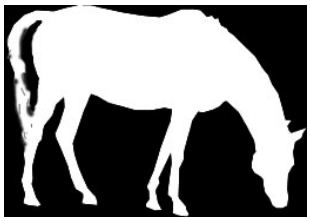
5. Use the wizard to add a Texture map to our current clipmap texture. Select the "Tree1.jpg" as the color image and render. You should now see a tree that has been clipped out and will cast detailed shadows.

Fade mapping

Realsoft 3D uses the term Fade in VSL to describe the transparency component that is cut out from an object. Just as we use the color channel to put an image or color onto an object, whatever is mapped to the fade channel will be cut out from the object. It is an advanced form of trim curve if you like - allowing you to cut out detailed or arbitrary edges to an object, and even give it levels of transparency where only half a cut is achieved.

The advantages of using fade mapping is we can get a soft edge - or antialiased edges to the cut, giving it that added level of realism.



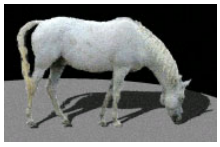


1. Create a new scene.

2. Create an analytical rectangle - any size. (Keep in mind the aspect ratio of the original image being mapped)

3. Create a new material, use a template. Select Fade map. Select the image "horse_alpha.jpg" from the tutorprojects\material\clipmap\textures directory. Drag this material onto the rectangle. Render the image.

Note how our horse is "back to front". we have mapped the image on the right to achieve the image on the left. Standard alpha uses white for solid and black for transparent. Realsoft 3D uses the opposite. Black is solid and White is transparent. Instead of going into photoshop and inverting the image, we can do this inside Realsoft 3D. In the materials Advanced tab :



4. Add a Constant above the fade texture. Change the Output of the constant (default color) to Surface:fade. Assign a value of 1 to the constant. On the fade texture map below our constant, change the operator to Subtract instead of Assign. This now takes the image alpha away from a constant of 1 or solid. This basically flips our alpha channel.

5. Render the image - our horse should be cut out correctly. But we still need to add the color information. Use the Texture map wizard and select "Horse.jpg". Render. You have just created a horse!

Targa clip maps

Targa image formats are the standard method for storing alpha information into one image. The procedure is exactly the same as the method outlined above. Only the same image is used for both color and alpha. When a Targa is selected as a Fade map, the Bindings tab must be set to use the alpha channel as the output to use for the fade channel. This is done by checking the Edit bindings button in our advanced VSL settings. Removing the "colorR -> Fade" binding, then selecting Alpha and Fade and Adding. This may not always be necessary - just check that the alpha is being used for the fade first. Often adding the texture first to a color map then changing the output to fade will set "alpha -> Fade" automatically. Inverting the alpha may depend on the source of the targa. If it is already inverted - you will not need to use the constant=1 value.

Scope mapping

Scope refers to how much of the current mapping will be mapped onto an object. We could map only text onto an object for example, by essentially clipping the background with our alpha and only the text will be mapped onto the object. But rather than actually CUTTING the object, we are cutting only the texture. This has powerful possibilities - especially for adding textures over already assigned textures! (e.g. adding windows over a brick wall.)

1. Create a new scene.

2. Create an analytical sphere - any size.

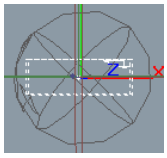
3. Create a new material, use a template. Select "color". Assign this material to the sphere by dragging it onto the object.

4. Create another new material, use a template. Select Scope map. Assign the texture "scope_alpha.jpg" to the scope field. The texture can be found in the tutorprojects\material\clipmap \textures directory. The adjacent image shows white text on a black background. This will give the white areas a scope of 1 and black areas a scope of 0.



5. Add a color gradient to our texture - use Texture map wizard and select "scope_coloring.jpg" from the tutorprojects\material\clipmap \textures directory.





6. Map this new texture onto our sphere using parallel mapping - make it roughly in the proportions to the image on the left.

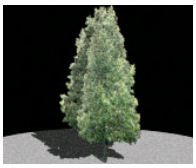
7. Render the image. We should have achieved a blue sphere (default color material) with the writing "scope mapping" written in color on the side of the sphere. Note how our color gradient was simply clipped by the scope mapping scope texture. We can also select Finite XYZ under the mappings Spec tab to restrict our texture from penetrating both sides of the sphere.

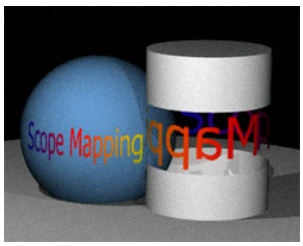
We can keep adding materials in this manner to get windows on walls, rust into crevices, writing on signs, etc.

Refer to "Adding Dirt" tutorial for further explanations in scope mapping.

Flat Image Vs 3D Object

One of the biggest challenges with any clip map is to have the image facing toward the camera at all times. We can tolerate a certain degree of movement - up to a point the illusion is ruined and the clip map is exposed for what it is - a single plane. We have certain "hacks" that we can utilize to give clip mapping an added degree of realism. One such example is to place the two planes at right angles to each other to form a "cross" in plane. Providing that the object (such as a tree) is roughly symmetrical, the illusion can be maintained from a 360 degree rotation.





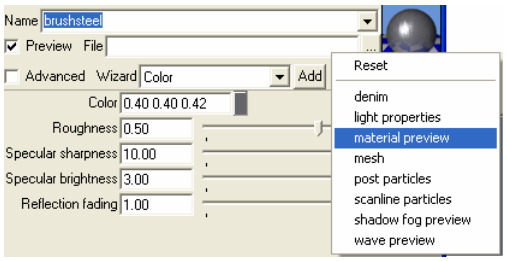
There are various mapping methods such as the Look at controller that can be used to make clip mapped objects follow the camera if needed.

But one need not be restricted to simple rectangular forms either - we can just as easily map a cylindrical map and clip out a 3D object.

Previewing Materials

When editing materials, you can preview the result by checking the Preview option of the property window. The preview window will be updated every time you change the material. The material icon is also automatically updated after each preview rendering.

The program uses a simple default scene for previewing the material. However, one single preview scene cannot show the characteristic properties of all possible materials. Therefore, the program provides an option for selecting the most suitable preview project for each material. The 'previews' folder contains a collection of predefined preview scenes. You can select the most suitable one using the File gadget. The name of the preview file is stored with the material when you save it to a disk file.

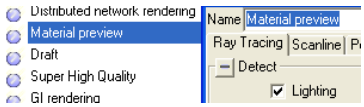


Selecting the preview scene

Note that the preview window is actually a small view window. You can zoom, rotate and pan it by dragging with the right mouse button Shift, Alt or Ctrl key held down. The window can be set back to the original scale and orientation using its Reset popup menu.

Changing rendering settings for the material preview window

The default collection of rendering settings contains an object called Material preview. You can control the rendering settings for the material preview by editing this object. Save the startup project after editing if you want to change the rendering settings permanently.



Editing rendering settings for the material preview

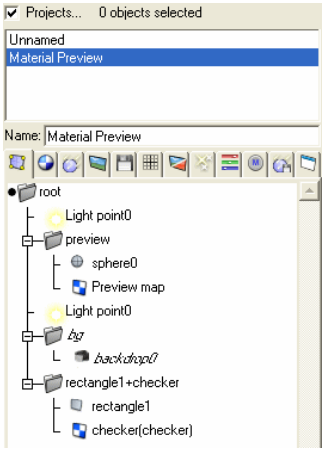
Post processing settings for the material preview window

Some frequently used material properties are rendered using the post processing system. Glow is a good example of this. Therefore, the material preview scene should include a valid post processing configuration.

You can define the configuration by including a post image with the suitable sub modules to the material preview scene. See the manual chapter Post Processing for further information. The name of the post image should be set to 'Material preview effects'.

Customizing the default preview project

The file 'material preview' included in the Previews folder defines the structure of the default preview scene. The scene is always loaded at program startup. You can find the scene in the Projects list of the select window and change it as desired. Save the edited file back to disk if you want to change it permanently.



The structure of a preview scene

The preview scene can include any objects and materials. However, it should always include one material mapped test object. The mapping object should refer to a material called as Preview. The image below shows the hierarchy of a typical preview scene.

The material preview system loads only objects, materials, post image effects and post particle effects from a preview file. Therefore, use of the Selected Sections option (with the above mentioned sections checked) when saving a preview project consumes less disk space.

The same guidelines are valid for building additional preview scenes, which can be saved to the Previews folder.

Combining 2D and 3D

This tutorial demonstrates advanced techniques for adding 3D objects into a real photographed scene. It shows various tricks for creating a seamless interaction between 2D and 3D worlds and makes frequent use of VSL. It is therefore intended for advanced users who understand the basics of programming languages.

Level: Advanced

Sample files: 'tutorprojects/material/lake/boat', 'tutorprojects/material/lake/Lake'

We will use the image lake.jpg found in the textures folder of the program installation. Go to the Images tab of the Select window, select New from the popup menu and double click the new image object to open the property window. Use the browse button of the Image file gadget ("..." button) to pick the lake.jpg file. The property window displays the image shown at the right.



A lake scene

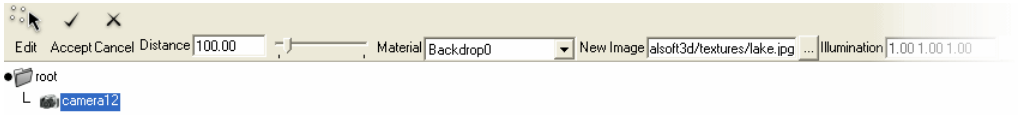
Our main objective of this exercise is to add a boat sailing across the lake. We want to see the reflection of the boat on the water (among other things) so that the animation looks convincing. We do not have exact information of the position of the camera or the dimensions of the lake. Therefore, we have to estimate their coordinates roughly to make things work well.

We start by setting up the view camera. Close the image property window. Reset the view window to the default front view (by clicking N button of view controls). Then open the view property window from "windows/view property window". Select the first tab Camera, which contains the information of view's internal camera. Select Perspective projection from it if not already selected.

We may think that the front bank of the lake is at the world origin, and the camera is approximately 10 meters behind it. The camera is held at a typical eye height - say, 1.5 meters above the ground. Therefore, enter the values '0 1,5 10' to the Position field. The camera is directed quite horizontally, therefore enter the values '0 1,5 0' to the Aimpoint field. The last important setting is Focal length, set 50 mm to that gadget. Now the camera is set and you may close the view property window.

We need to store the defined camera orientation more permanently. Therefore, go to the Creation tab of the toolbar and click Camera tool icon, and select Accept. This creates a camera object that has the same settings as the view camera.

The next step is to create a backdrop plane that shows the photograph in rendering. Make sure that the created camera object is selected in the object hierarchy. The tool control bar automatically displays the camera related tools. Look for Backdrop. Click the button, and new controls are shown. The first, Distance, defines the distance between the camera position and the backdrop plane. Set 100 meters, which is approximately the distance across the (quite small) lake. Leave the next gadget, Material, empty; we have not defined any material for the backdrop. The third gadget New image can be used for defining a texture map for the backdrop. Click the browse button beside it ("...") and select the image lake.jpg from the Textures folder. The last Color gadget won't be used, because the color comes from a texture map. Click Accept.

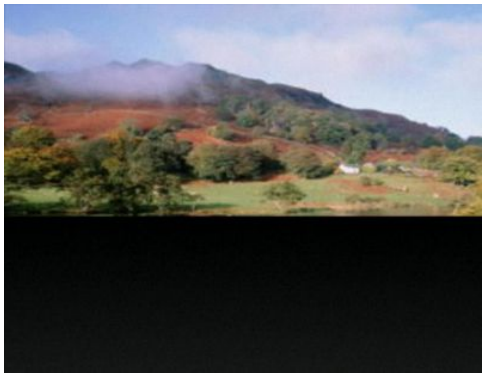


Creating a backdrop

After these steps, a backdrop plane and a texture map to a suitable material becomes inserted into the camera object's sub hierarchy. Render the view and you will see the lake image filling the view. The backdrop plane has such a special property that it automatically expands itself to fill the whole view of its parent camera object.

Next we will add a plane representing the water surface of the lake. The backdrop plane already shows the water, but it is in a wrong place (100 meters away), and is oriented vertically, so it cannot interact with objects close to the camera. Therefore, take a top view. Switch to Parallel mode by disabling View/Camera/Perspective from the view popup menu, for example. Then zoom right out until you see the backdrop plane. (Drag the root directory to the view window will set clipping planes to show all objects.) It may be wise to open the numerical window (View/NumericWindow in the main menu bar) if not already opened. Select the Analytic rectangle from the rectangle tool's shortcut menu in the Creation toolbar section, and draw a large rectangle covering the view. Note that the edge of the rectangle should go behind the backdrop plane (z coordinate over -100 meters).

When the rectangle is ready, close the numeric window. You may rename the rectangle as Lake for easier identification. Then select the camera object and pick the menu View/Camera/Object view (Right click on the view window) from the view's popup. This sets the view window to show the orientation stored in the camera object. Render the view. The created ground plane hides the backdrop's lower half. Optimally, the rectangle surface should become visible about at the waterline behind the lake. If this is not the case (for example, no water is visible at all), select the camera object, open the property window, and go to the Spec tab. This shows the camera specific object attributes. Change the Aimpoint from '0 1,5 0' to '0 1,7 0', which directs the camera slightly upwards, therefore lowering the ground level visible through it.

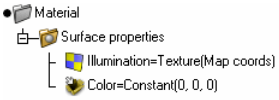


Water surface inserted

Update possible camera object changes to the view by selecting View/Camera/Object view, and render (Dragging the camera from the Select window to the view will achieve the same result). When the adjustment is suitable (in the image above, some water is slightly visible at the right edge), close the property window.

Now we have a rough model of the water. The next step is to make a material for it. Go to the Materials tab of the Select window. You may get rid of extra materials by selecting Materials/Purge Material Library from the main menu. The only material that is left is called backdrop0, which was created by the backdrop tool. Select it and select Duplicate from the popup menu. Double click the icon of the copy to open the property window. Rename the copied material as lakesurface in the Name field. We do not call it water, because the created material will not have the optical properties of water, but it will include quite a special combination of shading effects.

Enable the Advanced option and maximise the VSL Objects frame of the property window. Drop a Constant VSL object onto the Surface properties shader. The inserted Color=constant(0,0,0) removes diffuse shading from the water. This is necessary because the lake photograph already includes such shading, and we do not want to shade it again in the 3D model. The first version of lakesurface is now ready and you may close the property window. Later, we will improve the material step by step to achieve better and better integration between the 2D photo and inserted 3D objects.



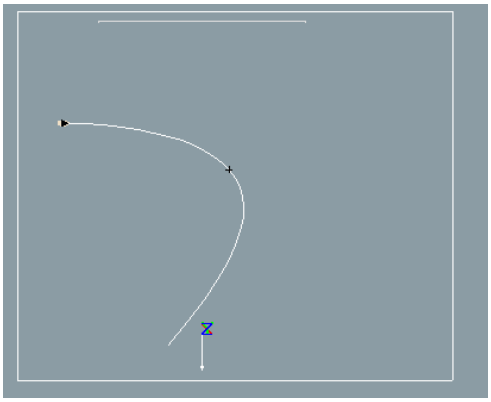
The first version of the material for the water surface

Go to the object hierarchy tab of the Select window and select the rectangle representing the lake. Select the Materials tab of the main toolbar and click the Default map tool icon. Select lakesurface into the displayed Material gadget in the tool controls, and set Mapping type to Camera. Click Accept. If you now render the view, the ground rectangle has disappeared and all you see is the lake photo. Anyway, now there is a real 3D ground plane in the scene and the color information for the water area comes from a sensible distance. Save the project as lake1 to a suitable temporary location using File/Save as menu to backup the work done so far.

Path animated boat

Next we add a small boat to the scene. Therefore, reset the view and select parallel projection. You may model the boat yourself, or load the included simple boat object, which was freeform modeled. Use Paste from file of the Select window's popup menu for this purpose. Remember to disable the replace option if you use File/Load menu for loading the boat, otherwise you loose the already modeled scene. The boat should have sensible dimensions: the length can be about 3-4 meters. You can use the numeric window and the scale tool to adjust the boat size to the right magnitude. Using the front view, move the boat so that its bottom is slightly below the waterline (y coordinate zero; world origin is at that level as well).

Take the top view and zoom right out, to see the whole lake rectangle (alternatively, drag the lake rectangle from the Select window into the view to zoom the object). Move the boat to the top left corner of the lake rectangle. Then select the curve creation tool from the toolbar and draw a cubic curve starting from the boat, going to the right and downwards, then left and down. In other words, in the camera view, the boat will appear from the left near the back shore of the lake, make a turn at the right and then pass the camera at a close distance disappearing to the left, like in the image below. Click Accept to finish the curve tool.



The path for the boat

Select the boat and the nurbs curve, in this order. Go to the Lattice mapping section of the toolbar and select the Path tool. The default options produce a directed path, which moves and rotates the boat, which is exactly what we want in this example, so just click Accept.

Open the AnimationWindow from the main menu bar's View menu, if not already open. Play the animation to see how the boat travels along the path. If camera tracking is on, the view window changes immediately as the animation playback starts to show the camera object's view. You can control this using the view property window's Camera/Tracking section. You can also use the View window's popup menu item Camera/Object view to take the camera view.

After the playback has stopped, edit and fine tune the nurbs curve from the top view if necessary. Then return to the camera view, move the time slider to a position where the boat is relatively close to the camera, and render. The bottom of the boat is partly hidden by the lake photo. This is the first step towards the 2D-3D blending effect we want. Currently, the boat seems to be hanging in the air.



The boat does not appear to float in the water because reflections are missing

This typical problem can be solved by adding a suitable shadow, or in this case, a reflection of the boat to the water. This task is a bit tricky because only the boat should be reflected. Other reflections are already included in the scene. Therefore, we need to use VSL for customized reflections.

Selective reflections

Switch to the Channels section of the Select window. Select New/Float channel from the popup menu. Rename the created channel object as boatchannel. Then make the object hierarchy tab of the Select window visible and double click the boat to select it and to open the property window. Select the Col tab of the property window, and pick the item boatchannel from the Surface properties/Attribute list. Enter the value 1.0 to the field below it. In other words, we mark the boat with the boatchannel property value of 1, other objects, like the backdrop plane, have the default value 0 and will be excluded.

Go to the Materials tab of the Select window, and select the lakesurface material. Drop Constant VSL object to the surface properties shader, use the popup menu to change Output to Surface:Reflection, and change the black constant value (0,0,0) to light gray (0.7 0.7 0.7). Render the image. Note that we must be rendering with at least "Reasonable quality" rendering mode to see reflections. If this is not the case, go to the rendering tab and drag "Reasonable quality" onto the view window and render again.

There is a reflection of the boat on the water, but things are not quite right:

- The water is too light and reflects the mountains twice
- The boat reflection contains color from the scene behind the boat, just like the boat was translucent in reflections

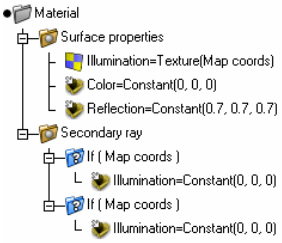


Incorrect reflections

Let's continue editing the lakesurface material on the property window. Drop a new Shader object from VSL Objects just under the Material node. Change the Type to Secondary ray. This shader controls traced rays - just what we need. Drop an If object into Secondary ray shader. Change Input 0 from the popup menu to Traced ray:boatchannel . The If object gadget at the bottom of the screen displays three conditions. Change the topmost X field to If inside and enter Lower=0.1, Upper=1.1 to the two numeric fields below it. Leave other conditions unmodified. In other words, we test here if the reflected ray contains nonzero boatchannel information.

Now drop a new Constant object into the If object. (Place it underneath and inside.) As explained above, this If block handles the boat reflection area, because it is executed only if the boatchannel value is between 0.1 and 1, and only the boat object has currently such a value (its boatchannel property was set to 1). The problem area is that the reflection also showed the background scenery. The reason is the original camera projection mapped illumination. Therefore, set output channel of the new constant shader to Surface:Illumination. The value zero is exactly what we want: we reset the illumination mapped from lake.jpg to zero before adding the reflection from traced ray to it.

Another problem was that outside the boat reflection, the lake reflected the scene behind it twice. The first correct reflection was already in the scene photograph. The second incorrect reflection comes from the added reflection property: rays reflect from the lake rectangle and hit the backdrop plane. This effect can be eliminated easily.



Select the If object and then Copy from the popup menu. Select the Secondary ray node from the VSL hierarchy and Paste from the popup menu. Then change the first condition of the new If object to If not inside. In other words, this VSL block will handle those reflections that don't have boatchannel value set.

Select the Constant object inside the If object and change its output to Traced ray:Illumination. This way we reset unnecessary reflections (not showing the boat) to zero. Test render to verify the result. This is also a suitable point to save your work again.



Correct reflections

Waves

The next effect to be added is waves around the boat. The main difficulty is that the lake photo already contains reflections of the scene in the background without any waves. Therefore, we have to somehow modify the existing reflections.

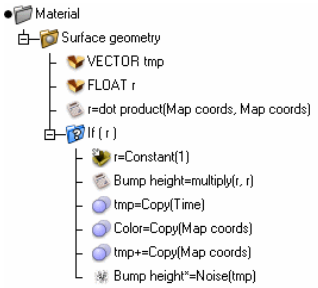
First we create a suitable procedural bumpmap for the waves. Go to the Materials tab of the Select window, and create a new VSL material. Open the property window (if not already open), and rename the material as waves. Open the advanced properties. Drop a Shader object onto the Material node, and change the type of created shader to Surface geometry. This shader is suitable for bump mapping purposes. Drop a Variable object to Surface geometry shader, change its name to tmp and type to Vector. Then drop another variable to the same shader, rename it as 'r' and set type to float. Then insert an Operation object to the shader, and select dot product from the long list in the Type gadget. Set Output to r variable using the popup. Both Map coords inputs are correct. This VSL object computes the square of the radial distance from the zero point of the mapping. We will use the radial distance to fade out the waves within a circular area around the boat.

Drop an If object to the shader, and set its Input 0 to float variable r with the popup menu. Change the first condition X from Always to If inside, and define Lower=0, Upper=1. The included code will be executed if the mapping coordinate measured distance (which can be adjusted by scaling the mapping object) will be less than one.

Drop a Constant object into the If object. Set its output to r, value to 1, and change Operator to Subtract from the popup menu. Then drop a new Operation object into the If object. Select multiply from the long Type list, Set Input 0 and Input 1 both to r, and set Output = Bump height. The purpose of these tedious and technical computations is to define a nice bell-like curve that fades out the wave smoothly. The nice and smooth curve used here is :

bump height = (r^2 -1)^2

We need a couple of VSL objects more. First add a Copy object to the end of the If sub hierarchy. Set Source to Surface:Time, and Destination to vector variable tmp. The Copy object's gadget shows three selectors which bind the sub channels. Change the first two to the None state, but set Sub channel 3 to Time. Then drop another Copy object to the If node. The default source Map coords is OK, but change Destination to tmp and Operator to Add from the popup. Now we have computed a set of modified mapping coordinates by adding the animation time to the third coordinate component. The purpose is to animate waves by sliding a noise field through the water surface. Sounds strange - but it works well! The last object to be added is Noise: Drop one to the end of the If hierarchy. Set output to Bump height and Input 0 to tmp. Set the Operator to Multiply from the popup menu. Then adjust the noise properties at the bottom of the property window: Set Octaves to 3 (it defines the level of detail in the waves), Amplitude = 0.1 (height of waves = 0.1 meters), Smooth and 1D enabled. The material is ready. Its structure is shown on the right.



The VSL structure of the wave material

Close the property window and rewind the animation to the beginning. Take a parallel top view and select the boat from the object hierarchy section of the Select window. Then select Autofocus to find the boat (Autofocus is in the view's popup menu and in the view control icons). Zoom out a bit so that you see a reasonable area around the boat. Select only the lake+lakesurface level from the Select window. Then go to the Materials tab of the main toolbar, and activate the Parallel map tool. Select the waves material from the material list in the tool controls.

Click first time just behind the boat (where the waves are supposedly highest), and move the mouse downwards and to the right (the "shift" modifier key held down) to get a uniform rectangle. Click a second time to finish the parallel map. The edge of the parallel map should be 1-2 times the length of the boat. You may resize the parallel map later to adjust the extent diameter of the waving area.

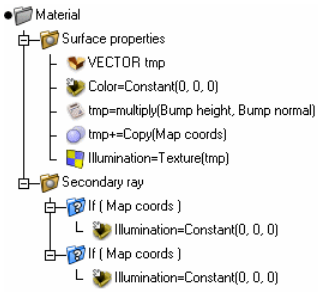
We have to animate the wave parallel map exactly the same way as we animated the boat. Make sure that the parallel mapping is selected. Move the mouse over the middle point of the object space handle on the view window, and drag the handle so that it is in the top left corner of the parallel mapping (just behind the boat). We do this to ensure that the maximally waving area follows the path, maintaining the originally defined 'behind the boat' status. The path tools fit the object space origin to the path. Press the Shift key down and multiselect the nurbs curve, which was used in animating the boat. Go to the Lattice Mapping toolbar section and use the Path tool to animate the parallel map with the default options.

Take the camera view and start animation play. You should see the parallel map following the boat across the lake. Stop the playback when the boat is quite close to the camera, and test render. You should see the boat reflection nicely twisted because of the waves. The strength of this effect can be adjusted by editing the Amplitude slider of the Noise object in the waves material hierarchy.

At this point, the original reflections of the photograph do not follow the defined local waves. Let's fix that next. Activate lakesurface from the material tab of the Select window and open the property window.

Drop a new Variable object to the top of the object list inside the Surface properties shader. Rename it as tmp and set the type to Vector. Then drop an Operation object just above Illumination=Texture object. Set Output to tmp, Input 0 to Surface:Bump height and Input 1 to Surface:Bump normal. Select Multiply from the Type gadget at the bottom of the property window.

Drop a Copy object just below the previously added multiply operation. Change Destination to tmp and Operator to Add from the popup. Then select the already existing Illumination=Texture(Map coords) object from the VSL hierarchy and change its input from Map coords to tmp. The material is now ready. The subtle effect of backdrop reflection waves is achieved by adding the bump vector to the texture coordinates directly! This is just a fake but better than static reflections.



The final version of lakesurface material

Foreground layers

The last part of this tutorial explains how to add foreground mask objects. Namely, the photograph includes a tree at the right edge of the screen, and also the front bank of the lake is visible. It may happen that the boat meets these areas while travelling on the lake and hides them, which certainly looks strange. Also, the front bank should not reflect the boat.

Close the property window and take the camera view. At this point we have to pay some attention to the image aspect. We will draw the mask objects over a backdrop image on the view window, and therefore the view must have exactly the same image width/height ratio as the camera object being used for the final rendering of the animation. The camera object was created from the view settings, so they may already match. The numerical window and other windows appearing at the bottom of the screen may have changed the view size, so you may have to close some of them to restore the original situation.

Another possibility is to use the Spec tab of the property window to edit the camera object's properties. In the Image aspect section, there are some predefined settings to choose from, or you may enter the desired custom value. The most suitable value depends on the source of the backdrop image. If the lake image was digitized from a PAL video signal, then we should use the same image aspect in the 3D model as well, otherwise the backdrop will stretch (spheres will look like eggs, etc.). Let's assume that the backdrop is a PAL image. Therefore, choose PAL, then close the object property window, open a view property window, select the leftmost Camera tab and open the Image Frame section. Then enable the Fixed size option and enter the same aspect value (0.75 for PAL) to the Image aspect field. The height of the view is adjusted accordingly. If the height grows, the bottom area of docked views becomes invisible, because the view cannot stretch. If that happens, reduce the width of the view by 50 pixels and enter the aspect again. Repeat this until the view is capable of displaying the image with correct proportions.

After these preparations, copy the camera view onto the view window by selecting the camera object and using the View/Camera/Object view popup menu function. If the preparations were successful, the wire frame of the camera is visible as a thin rectangular border matching the view edges. If the camera view rectangle is not visible, zoom out a bit. If the side edges of the camera view rectangle are visible but the top and bottom edges are not, or if the top and bottom edges are far away from view borders, the view - camera aspect matching has failed and you should recheck the aspect ratios.

After taking the camera view successfully, close the object property window and the view property window. Select the Images tab from the Select window. Drag and drop the lake image object onto the view. This activates view's continuous backdrop drawing. Select the curve tool from the toolbar. Set the Type field of the tool controls to Analytic, which option creates a filled polygon.

Start drawing a polygon from the bottom left edge of the view, so that the polygon follows the outline of the front bank. After the rightmost point of the bank, click one point at bottom right corner of the view, then at the bottom left corner, and then Accept to finish the tool (see the image). Draw another polygon to cover the tree at the right edge of the image.



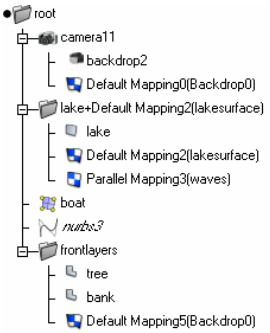
Foreground polygons

When the polygons are ready, open the view property window, go to the Draw tab and clear the Backdrop/Image field displaying the name of the image object, which was dropped onto the view. The backdrop disappears. Close the view property window.

Select the two polygons, and use the Drop to a level tool of the Select window's popup menu. Select and rename the created level as frontlayers. Open the property window and change the color of the frontlayers level to black. This eliminates diffuse shading (black diffuse color was added to the lakesurface material, but we will now use Backdrop0 material which does not eliminate diffuse color). Close the property window.

Make sure that frontlayers is still selected and activate the Default mapping tool from the Materials section of the toolbar. Select the Backdrop0 material, set Mapping type to Camera and click Accept. A default mapping object texturing the polygons becomes inserted to the hierarchy.

The long tutorial example is now ready. You can save and test render it, for example, to an .avi file, to see the created effects animated.





The final hierarchy and a sample image

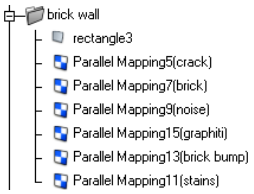
Adding Dirt

Adding Dirt

This tutorial outlines various methods of mapping your models to look realistic - in essence, dirtying up the otherwise shiny "super real" look of computer generated imagery.

Level: Medium

Perhaps the most useful method of creating additional detail to your scenes - scope mapping allows the user to add additional materials to a single object/level. Each successive material takes precedence over the previous material. e.g. the material at the bottom of the select window is placed on TOP of the material underneath in the model. (i.e. each material is layered successively, the lower in the select window, the closer to the top or front of the material.)



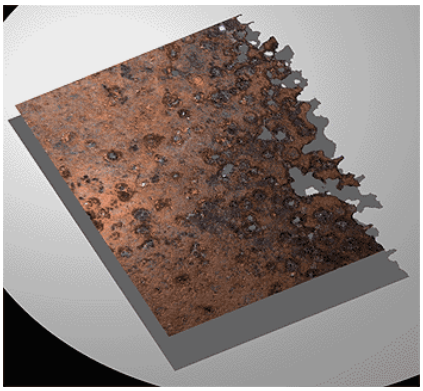
This is useful for adding graffiti to walls - rust stains, drains in the ground - or overlay any object or material. To effectively use scope mapped materials - one generally makes use of an alpha channel or seperate image that is used to 'cut' the material (unless a simple square parallel map suffices). This alpha channel can be applied to:

- The **Scope** channel - which will simply cut out the **extent** of the mapping.
- Or the **Fade** channel - which actually cuts the **object** that is mapped.

Obviously we can map any image to any channel in Realsoft 3D - but for the purposes of this exercise we shall deal with scope and fade mapping. To achieve the "cutting" action can also be brought about by a variety of means:

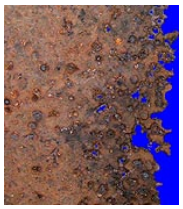
- Using the **clip** material in the standard /image templates
- Using an Image Alpha channel like in a 32 bit Targa file
- Or having a seperate Greyscale image to the color image.

Fade mapping with the CLIP function



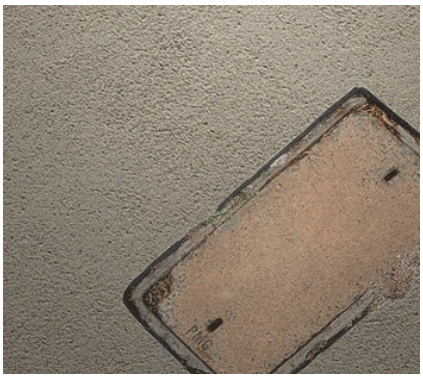
The clip material uses a color range to determine what extent should be cut. This material is best used for areas of solid color. If the mapped material has a blue background, we can set the clipping range to encompass all the blues to cut out. Refer to:

example project file: dirt/projects/iron.r3d



In this scene, the blue in the image is used as a mask. We can set any color as a range - ideally it would be a color that is not used in the image. This method generally produces a hard edge and is probably the least desirable.

Scope mapping with two images

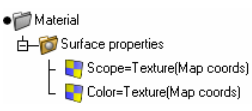


The next scene example project file is: dirt/projects/scope01.r3d.

This example uses two separate images. The diffuse color image D_manhole.jpg and an Alpha image (Cutout) called A_manhole.jpg. Its often a good idea to prefix all your textures with B_ for Bump, A_ for Alpha, etc.

This simple scene has mapped two materials onto one object. The foundation is the concrete material that surrounds the whole floor. On TOP of that (i.e. below it in the select window) we have put our manhole texture. Note the structure of the manhole texture.

It is very simple : The Scope channel is set to our alpha channel image (shown above). Note that black designates complete transparency - and white designates solid areas of color. Any grey between these values would give a soft transition between the two.



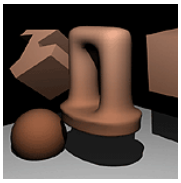
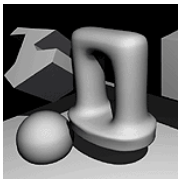
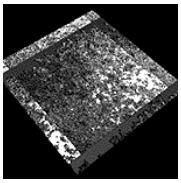
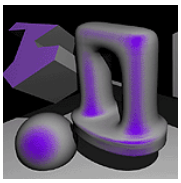


The color channel is set to the color image of the manhole. Note that the color image does not need to be cut out. The alpha image or channel does this for us.

The end result is a manhole sitting in concrete on one object. The scope channel is very useful for adding all manner of materials on top - Windows, graffiti, leaves, grime, etc.

Dripping rust often congregates under beams and appendages - adding a simple scope rust material over and over again can add whole dimensions to your scene - all from either an image map or simply a dark color mapped on top of another material with 0.25 % scope.

Materials Useful for Dirty Mapping

	**Adding Noise over materials** Using a low scope - one can "dirty" otherwise bland areas or disguise repetitive materials by simply adding a noise texture.
	**Faking shadows** Using a simple greyscale gradient image, we can add shadows under eaves, or add subtle radiosity like effects to objects. Adjust the scope level to increase or decrease the effect of the material. project: dirt/gradient.r3d
	**Angle based scope** This material takes into account an objects ANGLE to the materal mapping - hence we can add "dust" to the top of objects simply by applying this material downward. Often having a white down material and a dark up material can effectively simulate both sun burn on top and grime underneath. project: dirt/dust.r3d
	**Darker bumps** This material uses the height of the bump to add grime into the gaps. It essentially only burns the darker regions of the bump material - but can be good for tiles and very rough materials. project: dirt/dirtybump.r3d
	**Perpendicular scope** This can be used to fine tune materials that should only fall on a certain area of an object - used in conjuction with a finite mapping method - this is particularly useful for additional dust and grime. project: dirt/smscope.r3d

Truly Effective materials make use of the whole gamut of available channels: Specularity, Color, Fade, Bump, Transparency, etc. Through clever use of materials we can keep geometry to a minimum and maximize the realism of a scene. Let's have a look at a scene created earlier and examine how it works.

Dirty Wall

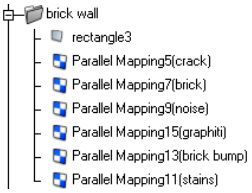
This example scene is a simple exercise in creating realism through materials. Note that there are only 3 rectangles and 1 cube in this scene! We could have cleaned up the junction between grass and edge with a NURB surface if desired.

example scene: dirt/projects/dirtywall.r3d

There are several elements in this scene: The wall, grass, downpipe and grass edge. Let's examine each in turn and how we achived each material.



The Wall



This object has the most materials applied. At the "back" is our first crack - this is simply a bump image (black and white) of a crack. Note if we map consecutive bumps to each other, the background must be black as white is high, black is low. We wanted the material to crack in, so we have used a negative value to "invert" the bump.

Next is the brick photo by itself. This is only a small section of a brick wall made tilable. This is best achieved in Photoshop by using the filter/offset with 'wrap around' set, to see how a material tiles. Using the stamp tool, you can make a material tile properly.

Over the brick we have put a subtle noise. This adds a basic black and white noise on top of the brick to give it a slightly dirty appearance. Note we use the Scope parameter of the parallel mapping object itself (under the Properties/spec tab) to give the noise more or less effect. Currently set quite low (0.6).

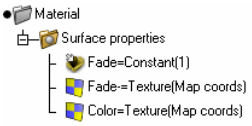
On top of the brick we have added a stain; vertical water stains that have dripped down from the top of the brickwork. This material is simply a scope material. The scope channel uses a B+W image again, with white designating the area of effect. Note we can put any material or color in this material and the area of scope limits it to this. In this material however, we have simply used a color- hence the scope limits the color to the area of the scope image, thus creating our vertical stains.

Then we have a simple brick bump. Note we could just as easily have added the bump with the brick material. We have used the original bump material for the bump channel, but ideally we would create a B+W bump material to emphasise the areas of interest (the mortar joints).

Last is the graffiti. Like our stain, the graffiti is simply a scope material. The white defines the area of color, and creates our graffiti. Note that any of these scope materials can be moved freely around the wall or duplicated to create more. Also, we have set the Finite options under the parallel mapping object. This limits the area of effect for this material within the boundaries set by the mapping object. Unless we wanted to tile this graffiti, generally we should use Finite mappings for scope based objects.

Grass foreground

This object was used to show the use of the fade mapping channel. We could have just as easily mapped this material onto the brick wall using the scope channel instead of the fade channel.



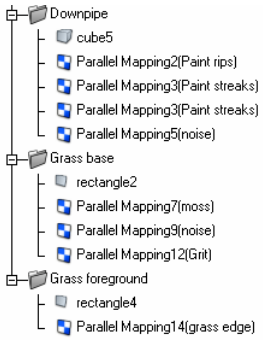
The grass foreground material uses the fade and color channels. Note that we have also used a fade constant. We set the fade as a constant of 1 (white) then with the image mapped to the fade channel, we use a subtract operator to essentially "invert" the alpha. We do this because Realsoft 3D uses white for transparent and black for solid in fade channels only. Scope uses the opposite. Hence, we invert our alpha channel by doing this.

The texture is a Targa, so the alpha is embedded into the same texture. Hence, both the color and fade textures have the same image applied. Under the Bindings tab of the fade texture, you can check that it is bound to alpha->fade. An alpha JPG would have something like ColorR->fade.

This material is then parallel mapped to the rectangle and repeated in one axis. The alpha acts as a cookie cutter and actually cuts the object itself (vs using a scope which merely limits the material).

Grass

This baseplane is simple: it has a grass texture parallel mapping tiled, then another 'Grit' material applied over that - essentially simulating some sticks over the grass - and then a duplicate copy of our Noise material from the brick to simulate a bit of un-evenness.



Downpipe

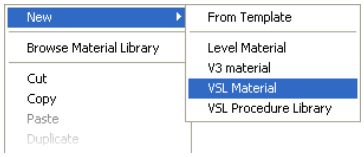
This object is probably more complicated than it needs to be - but shows a multiple layering of scoped materials. Note these materials make use of the scope - hence we have not globally set the color of the object itself - hence the color is derived from the object properties/col tab. The multiple scoped materials give a random appearance to flaking paint - all 3 of which are simply B+W scope maps and a color. We then added our noise again to give it some random discolouration.

Color Control

Creating 3D rendered images often requires the use of textures, like texture maps and bump maps. Consequently, sooner or later you may want to change the color or the brightness of the rendered texture a little. One way of doing this is adjusting your textures in your preferred 2D program and reload the modified textures. There is also another option: you can adjust colors and brightness with the use of the VSL Curve object in Realsoft 3D directly, which is much more intuitive and also a lot faster.

This tutorial chapter shows how to control the color and brightness of your texture by using the VSL Curve object.

Level: Beginner/Medium

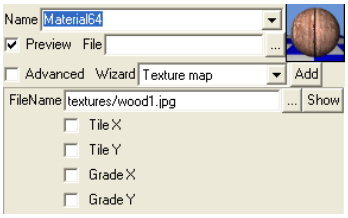


First we start by creating a new material. In the Materials tab of the select window, click the right mouse button (RMB) and select New/VSL material.

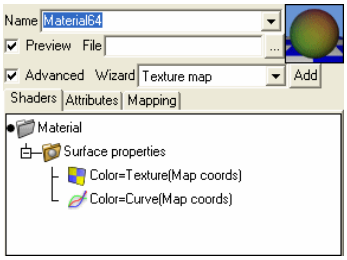
This adds a new and empty material, called Materialxx, to your library. If you have material previews activated - at the Materials tab right click and select Show Previews, a black square is added at the bottom of the materials list. Double click the new material to open its property window. You can also press the 'p' key to open the property window.

At the top of the property window, the name of your material is displayed. Rename it to "ColorWood".

In the property window, turn on the Preview checkbox, so you can immediately see the result of your actions. A white sphere appears. Now use the wizard to add a "texture map". Select wood1.iff from the textures directory and as soon as you load it, the white sphere in the small preview window changes into a wooden sphere.



The white sphere in the preview window has changed into a wooden sphere

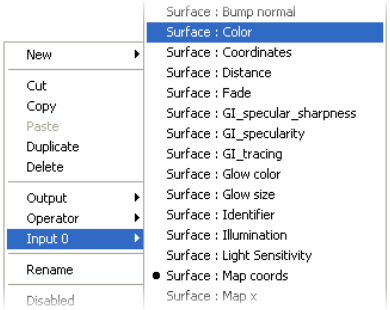


Adding the Curve object changes the sphere into a colorful object!

Next, check the Advanced checkbox to get access to the VSL components of the property window. From the VSL Objects, select the "Curve" object and drag-and-drop that in the Surface properties shader. Make sure the Curve object is the last object in the VSL hierarchy as shown below:

The effect of adding the Curve object is that the color of the sphere in the small preview window changes immediately from a wooden sphere into a colored sphere. Let's fix that. The Input0 variable of the Curve is by default set to Surface:Map coords. Select the Curve object in the VSL hierarchy and open the RMB pop-up menu. Select Input0 from the pop-up list and notice the black dot at Surface: Map coords. Now change that to Surface:Color (simply select that from the list). Immediately the appearance of the preview sphere is restored to a wooden sphere.

The material is now ready, so you can close the material property window.



By default Input0 is set to Surface:Map Coords; change that to Surface:Color

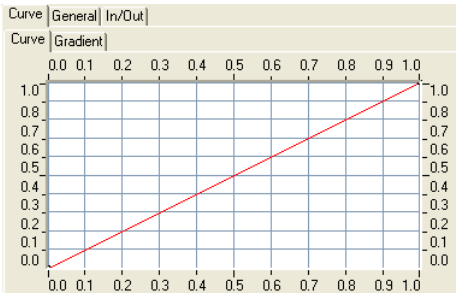
Setting up the scene

For this tutorial we need a very simple scene. In front view, create an analytic sphere and add a point light source somewhere above the sphere, so the scene is lit. This will do just fine. Make sure the sphere is selected in the select window and go to the materials tab. Select the ColorWood material, press the RMB and select Map -> Parallel from the pop-up menu. In the view window create a parallel mapping that encloses the sphere. Render the view to check your results.

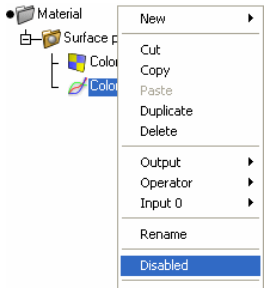
Using a curve to change color

Controlling the color and brightness of the woodtexture is now a matter of modifying a curve. So, let's have a closer look at this curve and how the curve controls color and brightness.

Make sure the property window for the ColorWood material is open and the Advanced option is set, so we have access to the VSL components of the material. Click on the Curve object and have a look at the lower part of the property window. The first thing you see is a red curve with two knotpoints; one knotpoint at (X=0, Y=0 - lower left) and one knotpoint at (X=1, Y=1 - top right).



One knotpoint has coordinates X=0, Y=0, the other has coordinates X=1, Y=1



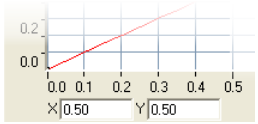
Enable and Disable the Curve object via the RMB pop-up menu

In its current state the curve has no effect on the material. To check this just render the view. Then right click on the Curve object in the VSL hierarchy, select Disabled and render again. Nothing changes. Now in the same way, enable the Curve object again.

The Curve object makes use of 3 curves, a red, a green and a blue curve, which in this case control the RGB color values of your texture. So, the red curve contols the R(ed) value, the green curve the G(reen) value and the blue curve the B(lue) value. We now only see a red curve because all three curves are exactly on top of each other, the red curve being the one on top.

To modify a curve, one clicks in the graph on the line to add a knotpoint, then move the knotpoint to the desired position. Let's try this and see what happens:

Add a knot point in the middle of the curve (X=0,5 and Y = 0,5) and drag that knotpoint to X=0,2 Y=0.8. Note that you can enter the exact coordinates for a selected knotpoint via the numeric input fields just below the graph.



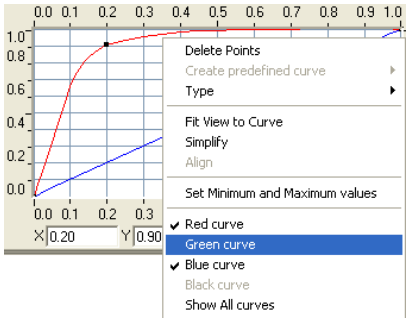
Numeric input fields for exact positioning of selected knotpoints

The first thing we notice is that only the red curve has moved. At it's previous position a green curve appears, which also has a knotpoint. If you move the knotpoint of the green curve as well, a blue curve appears, also having a knotpoint at the same position. When you create a knotpoint while all three curves are on top of each other, all three curves get one at the same position. Because the material preview in the property window is active, you can immediately see the result of your changes. Render the view and you will see that the material has now a more orange color.

The conclusion is that by modifying a curve, we can influence the color of the texture. High values in the Y-axis result in a higher RGB color value. Because we moved a knotpoint in the red curve to a higher Y-value, we increased the value of the red color component of the texture. Therefore the texture appeared more reddish. Similarly, if we move this knotpoint to a low Y-value, for example to X=0,8 Y=0,2, the amount of red is decreased and the texture will become greenish.

Of course you can also decide to move for example only the green curve, or both red and blue. Just try some of these variations and see what happens. Remember that undo (ctrl+z) restores the curve to its previous shape. If you get completely lost, drag select all knotpoints you have added and select Delete Points from the RMB pop-up menu. This will restore the curve to its original shape.

By default all three curves are visible and editable in the graph. Sometimes it is much easier if you have only one curve or two curves visible in the graph. If you want to exclude a curve from being displayed in the graph, open the RMB pop-up menu and deselect the curve. Only the visible curves in the graph can be edited.

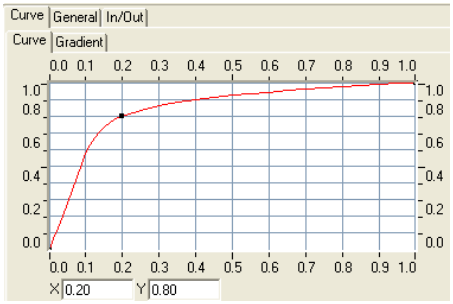


The green curve is excluded from being displayed in the graph

When you move a knotpoint of a single curve, you control the respective R,G or B value. What would happen if you move a knotpoint in all three curves at the same time? In that case all RGB values are changed in the same amount and thus will the color get lighter (if the Y-value increases) or darker (if the Y-value decreases). Let's see how this works.

Make sure all curves are enabled. Delete all knotpoints you have added, so the curves are restored to their original shape. Again add a knotpoint at X=0.5, Y=0.5. To move a knotpoint for all three curves at the same time drag-select the knotpoint (this selects the knotpoints for the red, green and blue curve simultaneously) and move it. Now move all three knotpoints to X=0,2 Y=0,8 and render the view.

As expected, the sphere now has a color which appears whiter. Now you can also predict what happens if you move the knotpoint to X=0,8 Y=0,2.



Knotpoint for all three curves moved to X=0,2 Y=0,8

Until now we have created and moved new knotpoints. You could also decide to change the position of the starting point and end point of curves. Move the starting point of the curve to X=0 Y=0,5 and move the end point to X=1 Y=0,5. What happens? The curve is now a straight horizontal line. All RGB values are made the same, which results in a shade of grey. Subsequently, if we move the horizontal curve upwards, the color will change more to white; move it down to change more to black. This shows how you can control the color of your objects and even overrule textures.



Chapter 8. Rendering

Introduction

The photorealistic rendering engine of Realsoft 3D uses scan line, ray tracing and post processing techniques to produce images with a high degree of realism.

In order to generate images, the rendering engine needs the following information:

- *A scene* containing geometric objects and materials.
- *Rendering Settings,* which control how the rendering engine performs its job. For example, rendering quality belongs to rendering settings.
- *Post Effects*, such as gamma correction, dithering and lens flare.

The default startup project defines reasonable default settings for photorealistic rendering. The only thing you have to do is to create a scene and click the Render button.

For example, to render a sphere:

1. Click the Analytic/Sphere tool and enter two points through the view window to define the position and the size for the sphere.

2. Click the Render button in the bottom of the view control bar.

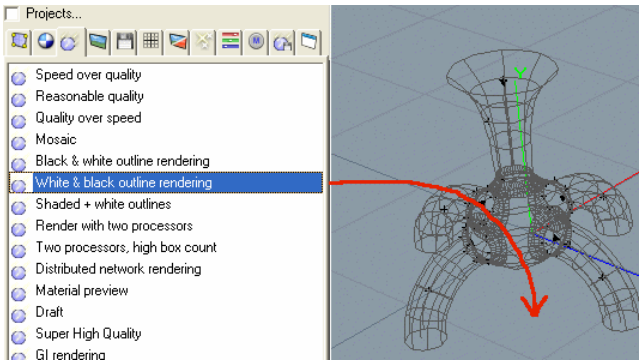
Specifying rendering settings for the view window

Rendering settings can be defined by dragging and dropping a rendering settings object into the view window. You can also select the rendering settings through the View Property window/Render tab.

For example, to activate Outline rendering:

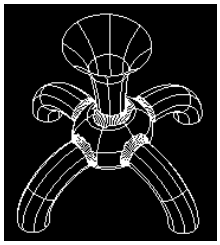
1. Go to the Select window's Rendering Settings tab. The default startup project contains several predefined rendering settings object.

2. Drag&drop one of the outline rendering settings into the view window.



Drag & drop a desired rendering settings object into the view window

3. Click the Render button and an outline image is rendered.



Outline rendering

Specifying post effects for the view window

A suitable post effect collection can be selected for the view window the same way as rendering settings i.e. using the drag&drop technique.

For example, to activate a star filter post effect:

1. Go to the Select window's Post Image Effects tab.

2. Drag & drop the 'Default effects + Star Filter' object into the view window.

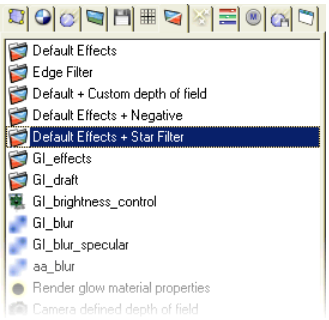


Image effects



Star filter

3. Render the view. Objects will be star filtered.

The view window of the standard configuration uses the Default Effects post effect configuration. It automatically renders glow, particles and depth of field, if the defined material properties, camera options, etc. so require. Therefore, you probably don't have to change the post effects configuration that often.



Note

The Post Image Effects tab contains two kinds of effects: Post images and actual effects applied to post images. In the example above we dragged & dropped a Post Image to the view window. Dropping individual effects to the view window adds them to the post image which is selected for the view. Effects can be rendered only by applying them to post images.

Grouping post effects to post images is explained in detail later in the post processing tutorials.

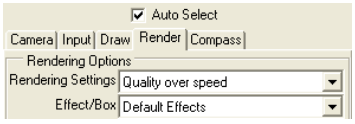
Rendering Settings

A *Rendering Settings* object is a collection of rendering options. Realsoft 3D's photorealistic rendering engine is extremely powerful and designed for maximum realism and image quality. Correspondingly, rendering settings include a large number of options for controlling the entire rendering process.

Rendering settings can be managed through the select window's Render Settings Tab. This tab shows the current library of predefined settings, and you can freely add new ones to suit your needs.

To create a new rendering settings object, select the New pop-up menu. This creates a new object into the select window. As usual, you can use the property window to modify properties of the newly created object.

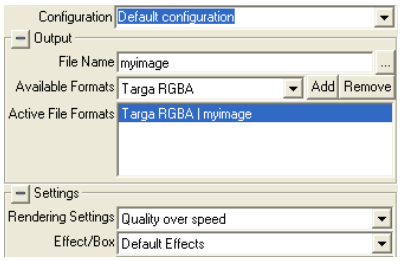
You can simply drag & drop a render settings object from the select window to a view window. Another solution is to use the view property window's Rendering tab to select the rendering settings. You can open this window from the View Properties menu, which is available in the view's popup menu (right click on view.)



Selecting rendering settings from the view property window

Rendering settings for file rendering

The File/Render window allows you to render the current project to a file. Rendering cannot be started before a rendering settings object is selected from the Rendering Settings list.



File rendering with Quality over speed

Rendering to a File

A project can be rendered to a file using various file formats, such as BMP, JPG, TGA or the native Realsoft 3D file format. The native format is the most flexible alternative, because it allows you to write any number of image channels to a file using any of the available data types: 8-bit byte, 16-bit word, 32-bit integer or 64-bit floating point.

Both still images and animations can be rendered. Animations can be rendered to a sequence of indexed images or to one (or more) AVI or FLC animation format file.

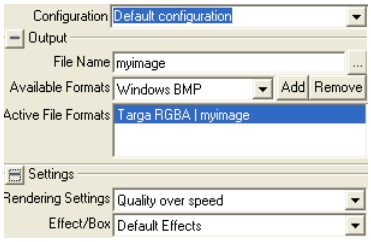
Rendering an image to a file

To render the current project to a still image file:

1. Make sure you have created a camera object to the current project. The file rendering tool needs one camera object to define the camera attributes such as the position and direction of the camera, focal length, etc.

Check that the camera object has a proper aspect ratio. Use the camera object property window to set the aspect to PAL or NTSC if you plan to render a video resolution image. If the purpose is to render a square-like image for a magazine advertisement, set the camera aspect to 1 (and use identical 'Width' and 'Height' in the resolution settings defined below).

2. Select the pull-down menu File/Render.This opens the file rendering window.



File rendering window

3. Select a suitable file rendering configuration. One of the predefined configurations, such as 'PAL', may already include suitable settings. If none of them is suitable, use 'Default configuration'.

4. Select a Rendering Settings object. Rendering settings define anti-aliasing quality and other rendering specific information. The default startup project includes a number of predefined rendering settings. You can create new rendering settings from the Select window's Rendering Settings tab.

5. Select a suitable post processing effect to the Effect/Box control. If you don't want to activate post processing, set this field to empty. The post processing system renders post effects such as lens flare and glow.

6. Configure Active File Formats. To do this, find the desired format from the Available Formats field and click the associated Add button. This adds the file format into the Active File Formats list. You can remove inserted formats using the RMB popup menu. Note that you can render to any number of files simultaneously. This is often a very useful feature. For example, you can render color information to one image and alpha information to another image.

7. Specify the resolution for rendering. You can also select one of the predefined resolutions from the 'Presets' control. Make sure that the resolution and the aspect ratio of the camera match. For example, if you plan to render a 'panoramic' image of width=2000, height=1000 for printing - the camera aspect should most likely be 1000/2000 = 0.5. Unfortunately dividing the height by the width does not always define a correct camera aspect ratio, because some display devices use non-square pixels. The usual video standards use non-square pixels. Fortunately you can find the appropriate resolution settings and aspect ratios from the presets.

8. Enter a File Name for the image.

9. Click the Render Image button to render the image.



Note

If the Render Image button remains disabled, make sure you have defined at least one image file format (animation formats cannot be used for still images) . Secondly, check that a proper file name is defined. Finally, check that a rendering settings object is selected.

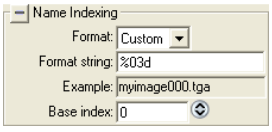
If you frequently need to render to certain file formats with a certain resolution and other settings, it is a good idea to create a new rendering configuration using the corresponding tab on the select window, define the preferred settings and save the new configuration to the startup file for later use.

Rendering an animation to a file

Animation rendering happens the same way as rendering still images. Instead of clicking the 'Render Image' button, click the 'Render Animation' button. When rendering an animation using an 'animation' format such as AVI or FLC, one file is generated (for example myanim.avi). Before rendering, the codec to be used is prompted - e.g. select "Cinepak" for avi - or use your discretion for format quality settings.

If an image format is used, the file render tool generates a sequence of separate images indexed with a frame number.

The Name Indexing controls define how the file name is generated from the actual file name and the current frame number. The default indexing format is flexible enough for most cases. If a more exotic indexing convention is desired, it can be achieved by selecting Custom to the Format field and entering a suitable string to the Format string field. The format string follows C language 'printf' conventions.



Indexing controls

Rendered images are named as follows: [filename][index].[extension]

For example: myimage000.bmp

The first portion is the file name defined by the user (such as d:\mypics\test). Index is the current frame plus the base index definedthrough the *'Base index'* field. The number of digits in the index portion is automatically extracted from the number of frames in the animation. For example, if the length of the animation is 120 frames, the index portion contains three digits. The extension is determined by the selected file format.

The *'Format String'* control shows the format string that is used for generating the index portion of the filename. If 'Format' is set to 'Custom', you can define an arbitrary format string to this gadget. For example, the string '%d' outputs indices without zero padding: frame0.tga, frame1.tga,..., frame99.tga... '%05d' would output: frame00001.tga. '%03d' outputs: frame001.tga, etc.

The '*Example'* field shows a complete sample filename generated by the current settings.



Note

Animation frame indices start from zero. If you want to start indexing from one (frame001.tga, frame002.tga...), set 'Base index' to one.

Box rendering

The 'Render Box' controls limit the rendering to a sub area of the total image. For example, it may happen that after spending one whole day for high-resolution rendering of a complex scene, the user notices that the color of a small object at the top left corner was wrong. Instead of re-rendering the whole scene, the user can render one quarter of the total image width and one quarter of the total height around the object to a new file. The position of the new sub image is known accurately and therefore it can be pasted over the original image using a paint or image processing program.

The controls are:

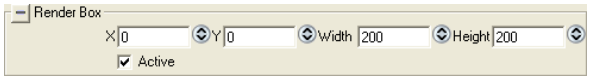
X - The horizontal position of the top left corner of the render box (measured in pixels)

Y - The vertical position of the top left corner of the render box

Width - The width of the box

Height - The height of the box

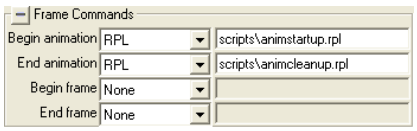
Active - Activates box rendering



Frame commands

The Frame Commands area contains four command gadgets. The commands can be defined using any of the installed scripting languages. In addition to this, DOS (or shell) scripts can also be used.

The following commands can be defined:



Begin animation

This command is executed once in the beginning of animation rendering. For example, a special hardware for animation recording can be turned on.

End animation

Executed once when the animation is ready. This can be used for clean up actions.

Begin Frame

Executed before rendering of each frame.

End Frame.

Executed after each rendered frame. For example, the rendered image can be recorded to a single frame recorder.

The frame command can access the file name via the following predefined macros:

> *[image]* - Full name of the rendered image, for example 'd:\images\test000.bmp'.*[path]* - Path portion of the file name without the last path separator, such as 'd:\images'.*[base]* - Base name (i.e. path and extension removed). For example 'test'.*[ext]* - Extension without the dot. For example 'bmp'.

For example, if the file name is d:\realsoft3d\images\test.bmp:

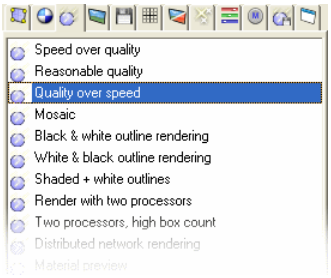
> [path] - results in 'd:\realsoft3d\images'
>
> [path]\[base].[ext] is the same as [image] - 'd:\realsoft3d\images\test.bmp'.
>
> [path]\[base].tga results 'd:\realsoft3d\images\test.tga'

Distributed Rendering

The distributed rendering system of Realsoft 3D subdivides an image to a number of sub images and uses available rendering servers to render them.

In a multi processor system, the program automatically examines the number of available processors and activates the respective number of rendering threads. The full power of the computer is used by default.

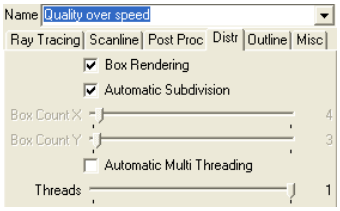
Sometimes it may be necessary to change the number of rendering threads. For example, you may want to leave one processor free for other tasks, or perhaps a complex scene requires so much memory that only one processor can be used.



Double click the desired rendering settings object

To change the number of rendering threads:

1.Go to the Rendering Settings tab in the Select window and double click the desired rendering settings object to open the property window.



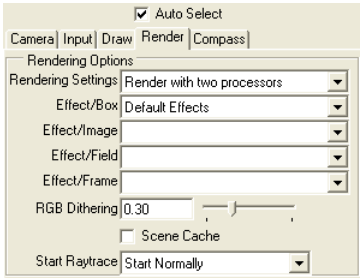
Set Threads to 1 to use a single processor

2. In the Distr tab, clear the Automatic Multi Threading option. Then set the Threads slider to a suitable value. For example, set Threads to 1 in order to use only one processors.

3. If you use multiple processors, make sure that Box Rendering is checked and that the number of boxes exceeds the number of processors. (Box Count X times Box Count Y >= Number of processors)

4. Drag & drop the rendering settings object from the select window into the view window. Click Render. If you use several threads, you can see how several render boxes become rendered simultaneously. If only one thread is used, boxes become rendered one after another. You can also select a suitable rendering settings object for the view using the view property window.

The file render window contains a similar rendering settings selector allowing you to use multiple processors to render images and animations to a file.



A view window uses 'Render with two processors' settings object for rendering

Configuring distributed network rendering

Distributed network rendering means using several workstations connected through a network for rendering.

Before you can use other workstations for rendering, the distributed network rendering system must be configured. The following things must be prepared for every workstation in the rendering network:

1. TCP/IP protocol must be installed and running. Unfortunately, this is something Realsoft 3D can't set up automatically for you. Consult your network administrator if necessary.

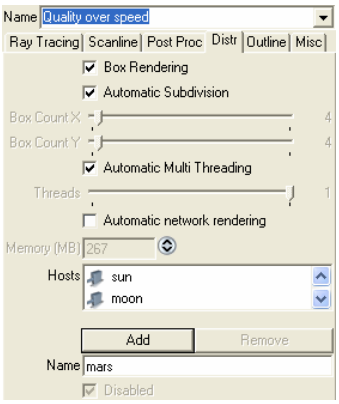
2. There must be a RendD rendering daemon (or the daemon version RendW which has a control window) running on every workstation that you want to use for rendering. You can install Realsoft 3D using the Network option to obtain a minimal render server configuration. The full program setup naturally contains all required components as well. RendD is a program, which listens to incoming rendering requests and spawns rendering server threads. It may be a good idea to insert RendD into the startup folders of render servers. Then the rendering daemon is automatically activated when a server machine is turned on.

Using network rendering

Once you have prepared the workstations for network rendering, you can ask Realsoft 3D to use one or more of them for rendering.

1. Define the host names you want to use for rendering in the Rendering Settings/Distr tab. If you have not defined the machines as "host" names then use the IP addresses instead. In a Windows XP network, you can use computer names (you can check and change the name in Control Panel/ System/Computer Name tab). For example: a machine is called 'sun' and sun's IP address is 10.0.0.5. You can use either the host name 'sun' or the IP address of 10.0.0.5.

2. Check and if necessary, adjust the render box count. By default, the program estimates a suitable box count automatically. By turning off the Automatic Subdivision option, you can change the box count using the Box Count X/Y sliders. The number of boxes should be higher than the sum of threads and servers so that a single image can be divided to all threads.



Network rendering setup. In addition to one local rendering thread, three servers are used for rendering: moon, mars and sun.

Optimal box count depends on several things, like the speed of the network, available memory, resolution of the image and performance of the rendering servers. If all the systems are equally fast, the optimal speed might be achieved by setting the box count to a multiple of the sum of threads and servers. In the example image above, such box counts are 4, 8, 12, etc. A high box count adapts better to speed differences, but requires slightly more managing work from the program. Finding the optimal configuration may require some experimentation.

3. Drag & drop the desired rendering settings object into the view window and render. You should now see several boxes being rendered simultaneously in the View window.

Symmetric multiprocessing and network rendering can be used simultaneously. In other words, you can use the processors of your local workstation as well as processors in other workstations of the TCP/IP network.

Rendering over internet

When using rendering servers over an internet connection, data transfering often becomes the bottleneck. The performance can be optimized by configuring a hierarchical rendering distribution tree. For example, consider a situation where you want to use your local rendering network and another remote network. The actual computers (their symbolic host names) may be as follows:

- *local_dp_1:* local dual processor system, the workstation you are working on

- *local_sp_2:* local single processor system, a render server
- *local_sp_3:* local single processor system, a render server
- *remote_sp_1:* remote single processor system, a render server
- *remote_dp_2:* remote dual processor system, a render server
- *remote_dp_3:* remote dual processor system, a render server

1. The remote network should be configured as follows:

- remote_sp_1: start RendD just with the default parameters
- remote_dp_2: start RendD with default parameters
- remote_dp_3: start 'RendD remote_sp_1 remote_dp_2'

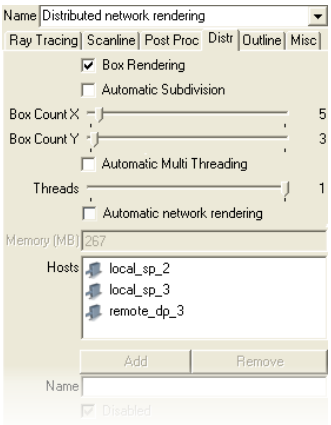
In above, the dual server (preferably the fastest of the remote servers) is started so that uses its both processors and also connects to the two other computers on the remote LAN.

2. The local render servers are started as follows:

- local_sp_2: start RendD with the default parameters
- local_sp_3: start RendD with the default parameters

3. The Distr tab render settings of the actual rendering client 'local_dp_1' are the following:

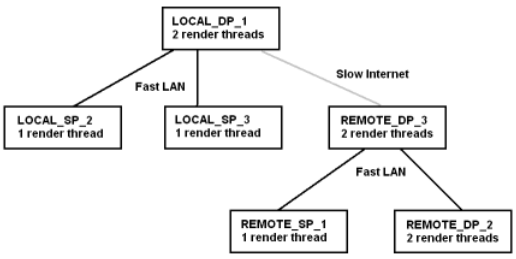
- Threads=2 (or Automatic Multithreading enabled), because there are two processors in the client itself to utilize
- The Hosts list is remote_dp_3, local_sp_2, local_sp_3



Now the rendering client uses 5 render engines (2 threads + 3 network servers), so the render box count should be at least 5. However, since the remote network with several processors is probably much faster than the local single processor render engines, it is best to use high box count that adapts to the speed differences. Multiplying the thread count by 4 might give a good result:

- Box Count X = 5
- Box Count Y = 4

Now, if the internet connection works fast enough, the remote network may render 6-8 boxes and the local network the remaining half of the boxes. The following diagram illustrates the structure of the rendering distribution:



For rendering farms, which have many workstations or whose configuration changes frequently, we recommend use of the Public Rendering Service. See the Reference Manual for details.

Lighting

A light source is a geometric object which radiates light of its own color. If a light object is black, it doesn't radiate light at all. A white object radiates all the main components (Red, Green, Blue) of light in the same amounts.

In real life, it is usually rather difficult to observe objects in lighting which have only one wavelength. Also, there isn't a material which would reflect only one wavelength. In Realsoft 3D, all this is possible, so you should be careful when selecting light source colors. For example, a totally red object isn't visible at all under blue lighting, because a red object doesn't reflect any blue light. A violet object looks red under yellow lighting.

Usually, it is advisable to create white light sources, so all objects will be rendered in their 'true' colors. Although the number of light sources is unlimited, they shouldn't be used indiscriminately. The time taken by rendering is greatly dependent on the number of light sources.

Light sources don't show directly in shaded pictures. So, if you want to see lamps or reflections of the light source on reflecting surfaces, you must put a cover made of, say, matte glass around the light source.

Unlike material mapping objects, light sources affect the whole scene regardless of their position in the object hierarchy.

Shadows on a floor

In the next example, we will test different light sources. The example includes a cylinder floating above a rectangle (floor), and some light sources casting shadows onto the floor (see the figure below).

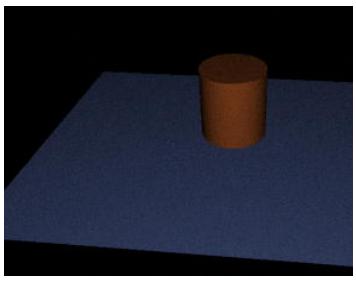
Tutorial level: Beginner

Example project: 'tutorprojects/rendering/lightsources'

1. Take a top view. Create an analytic rectangle (the floor), almost filling the whole view.

2. Create an analytic cylinder in the middle of the rectangle.

3. Go to the front view. The objects lie at the same level. Select the rectangle and move it to the bottom, then select the cylinder and move it above the rectangle. You may also change the colors of the two objects if you like.



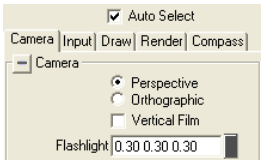
The first rendering of the test scene

4. Rotate the view upwards. Hit the '0' (zero) hot key to render the view.

The camera flashlight

The rendered image above shows that objects float in empty black space. However, we can see them - where does the required light come from? The answer is from the camera that took the picture. All camera objects, including view window's internal camera, include a flashlight.

The flashlight is quite a handy feature: it makes objects visible without actual light sources and also adds a suitable fill light to your images. Flashlight comes from behind the camera and therefore illuminates every visible place of the scene. To adjust the flashlight:



Changing the flashlight color

5. Select View Properties.. from the view window's popup menu. Go to the Camera tab of the view property window and enter a new value '0.3 0.3 0.3' to the Flashlight field. Then close the view property window.



Note

This changes the flashlight of the view window. In file rendering, the flashlight comes from the current camera object. Therefore, when preparing a real project for file rendering, remember to

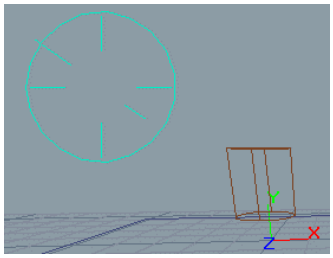
assign flashlight changes to the camera object after finding a suitable brightness level using view window rendering tests.

Now the flashlight is quite dim and we need some light sources to see the objects properly.

Adding a point light source

6. Take a front view. Go to the Light Sources tool tab and click the Point icon. Then click on the top and left side of the cylinder in the view window. When you now move the mouse, you can see a radiating circle. The radius of this circle will later define the intensity falloff of the light. Move the mouse until the circle includes the cylinder and most of the floor. Then click again to finish the creation.

7. Adjust the viewing angle and render the view.

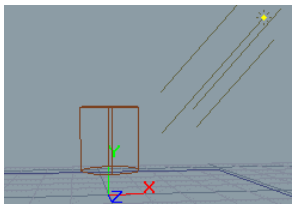


Creating a point light source

Now the point light casts a shadow of the cylinder to the floor. The shadow is 'perspective' - its width grows towards the top of the cylinder. There is another kind of light source available, which sends light from infinitely far away - namely the distant light source. All light rays from a distant light source are parallel and therefore its shadows do not have any perspective distortion. Let's experiment with the distant light source next.

The distant light

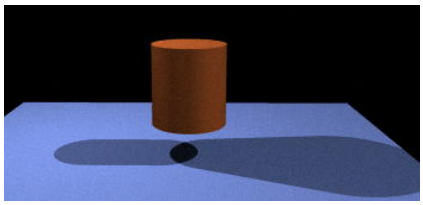
8. Take a front view. Activate the Distant light source tool from the Light Sources tool tab. Define the direction of distant light ray with two clicks on a view window. For example, draw the direction line down and left as shown in the example image.



A distant light source added

Note, that the length of the axis is not important, nor the position of it. You can even place the distant light source below the floor, and it still illuminates the upper side of the floor! The *only* relevant geometric property is the direction of the rays.

9. Render the view again (e.g. hit the zero key).



Shadows from a point light and a distant light

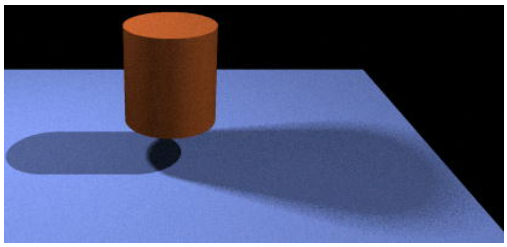
Soft shadows

Now you should see two shadows of the cylinder on the floor. The edges of the shadows from both lights appear to be sharp. Do this to see a soft shadow:

10. Select the Light point object from the select window. Open the property window and go to the Spec tab. Here you can see the point light specific object attributes. Enter a non-zero value to the Size field, for example 0.05 meters.

Note that five centimeters is quite a realistic size for a light source. The bigger the size, the smoother the shadow boundary becomes. On the other hand, the distance from a light source to the target also affects the result - the sun, despite of its huge size, casts sharp shadows, because it is so far away from the earth.

11. Set the Quality slider to the level 6. This slider controls the rendering accuracy of the light source: the higher the value, the better the rendering quality. However, improved quality means longer rendering times. If the smooth area on the shadow edge is wide, the quality setting has to be relatively high. Only slightly smooth shadows can be computed using lower quality. Usually values from 3 to 10 are suitable.



A sharp and a smooth shadow

12. Render the view.

This time one of the shadows has a smooth edge. Rendering is slower, but probably the extra realism is worth the time penalty.

Tip: Using a high level of anti-aliasing (especially stochastic anti-aliasing) also increases shadow quality. Therefore, the point light quality can often be relatively low in final high-quality rendering.

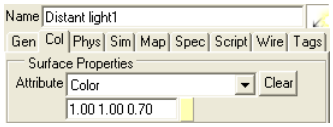
Color and intensity

So far, we have not paid any attention to the brightness of the inserted light sources. The brightness is defined by the usual object color and so called *Intensity* which is a light source specific property. The total brightness is a combination of these two:

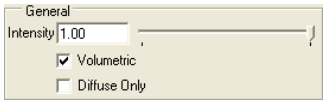
Intensity=0.5, Color=(1, 1, 1) is equal to Intensity=1, Color=(0.5, 0.5, 0.5).

We can test the effect of these attributes in our sample scene:

13. Select the Distant Light object. Open the property window and go to the Col tab. Since distant lights usually simulate sun light which is slightly yellow, change the Color of the distant light to pale yellow (1, 1, 0.7).



Changing the color of a light source



Defining light intensity

14. Go to the Spec tab. Since sunlight is quite bright, increase the Intensity from the default value 0.5 to 1.0.

15. Render the view.

Since the distant light was only slightly yellow, it can be difficult to see the effect of the color change in our colored test objects. Compare the color inside shadows - that shows the effect best. Remember that when trying to achieve a certain atmosphere in an artificial world, even subtle changes can be significant.



Note

The intensity value is not limited to 1. In the real world, the brightness of objects can vary enormously - the sun is millions of times brighter than a candle flame. Realsoft 3D is designed to deal with such differences. However, computer monitors or printed images can not duplicate this kind of phenomena. Therefore, use high intensity light sources with care: The result can be an over-exposed image showing plain white.

Controlling the falloff

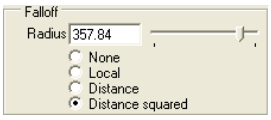
The two light sources of our test scene illuminate the whole universe. If you insert a new object 100 kilometers away from the cylinder, it is as brightly illuminated as the cylinder. This makes sense for the distant light source, which is almost infinitely far away. However, it is quite a strange property for a point light, which usually simulates light sources such as a light bulb.

Let's enhance the strange behavior of the point light:

16. Select the point light source, open the property window and go to the Spec tab. Increase the Intensity value to 1.0. Then render the view to see the result.

Let us analyze the result a bit: The floor now looks almost white. The color of the floor remains the same, but it is becoming so brightly illuminated that our 'film' (=computer monitor) gets over-exposed. The right edge of the floor is slightly darker - not because the point light is weaker there, but because light rays from the point light hit the floor at a smaller angle. We can compensate for the increased lighting intensity by selecting a suitable *light falloff* distribution for the point light.

17. The point light being selected, go to the Spec tab of the property window. The Falloff part of the window contains a falloff type selector. Select Distance squared. This means that intensity of the light decreases by the square of the distance - a point twice as far as the cylinder will receive only one fourth of the light.

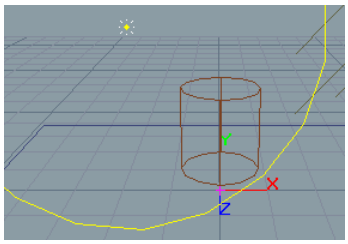


Select 'Distance Squared' falloff

This kind of falloff is physically correct for very small point-like light sources. If the light Size is greater than zero, Distance can be give a more realistic result.

Now render the view. The extreme brightness of the floor should now happen only close to the point light source. The bright area can be shrunk further by decreasing the Fallof radius of the point light. The initial falloff radius was defined by the circle shown by the point light tool (see the step 6 above). You can now change the radius using the controls in the Spec tab, or interactively as follows:

18. Make sure that the point light is selected, and hit the spacebar to enter the Edit mode. The point light source now shows a yellow circle in the view window. Move the mouse pointer over the circle so that the pointer changes. Then drag the circle towards the actual position of the light source, until the circle is about one half of the original size. Release the mouse pointer and hit the spacebar to exit the edit mode.


Editing the falloff radius of a point light

In the case of Distance and Distance squared falloff types, the falloff radius defines the distance where the light intensity is halved. The yellow circle edited above shows quite intuitively how the light decreases in your scene.

19. Render the view. The image brightness has returned to a suitable level.



Note

In Local falloff type, the circle shows the distance after which light intensity is zero.



Note

The distant light source does not (and can not) have Falloff controls. The intensity is constant everywhere.

Ambient light

The two shadows overlap slightly in the example scene. The area of overlap appears very dark, because only the weak flashlight illuminates it. If the viewing angle to the floor is small, the effect of the flashlight almost disappears.

The *Ambient* light source can be used to add a uniform base lighting everywhere in the scene. Ambient light simulates light that scatters from the walls of a room, from clouds, etc. - such light does not have any specific direction and it illuminates all parts of the scene.



Ambient light

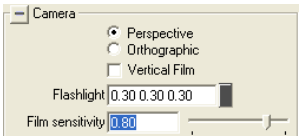
20. Click the Ambient tool of the Light Sources tool tab. Then click two points on the view window. The position and the falloff radius are not important this time (an ambient light can also have a falloff distribution, but we will not use this feature in this tutorial).

21. The default intensity of the ambient light is also 0.5, which is quite high, since we already have many light sources. Change the Intensity of the ambient light to 0.2 in the Spec tab of the property window. Render the view - the shadowed areas are now better illuminated.

Tip: one or two point light sources with 'Cast Shadow' option disabled often adds a more interesting base lighting to the scene than a single plain ambient light.

Controlling the overall brightness of the image

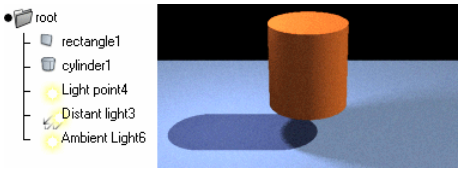
At this point, the image is again quite bright. What if we want to decrease the brightness? Do we have to laboriously edit the intensity of each light source? Fortunately, the answer is no - there is an easier way. We can change the sensitivity of the film:



Decreasing Film sensitivity

22. Open the view property window from the view's popup menu, and go to the Camera tab. Decrease the Film sensitivity value from 1 to 0.8.

A new test render shows that the image is now 20 % darker than before.



The final result

The special light

The special light source object allows use of light sources of an arbitrary shape. This light source is a level object, which turns all sub objects into light sources. The sub objects can be any geometric objects such as NURBS curves or meshes.

Tutorial level: Medium

Example project: 'tutorprojects/rendering/speciallight'

1. To get started, model the cylinder-floor scene of the previous tutorial without the light sources, or delete the three light sources after finishing the previous example.

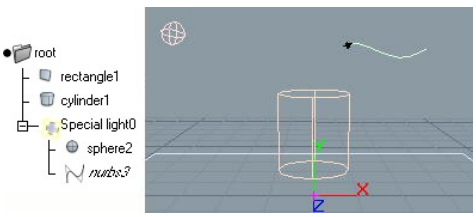
2. Click the Special light tool of the Light Sources tool tab.



3. Switch to the Analytic tab and create an analytic sphere to the left side and above the cylinder. Open the property window and change the color of the sphere to a pale red: '1.0 0.5 0.5'.

4. Create a NURBS curve to the top right side of the cylinder. Using the property window, change the curve to a pale green: '0.5 1 0.5'. Then go to the Gen tab and clear the Invisible in photorealistic rendering option. This is important - otherwise the curve will be ignored in ray tracing!

5. Select the sphere and the curve and drop them into the sub hierarchy of the Special light object.



The structure of the special light example

The object hierarchy and geometry should look as in the example image.

6. Select the special light source and go to the Spec tab of the property window. Since we have only one light source included, increase the Intensity from 0.5 to 1.0.

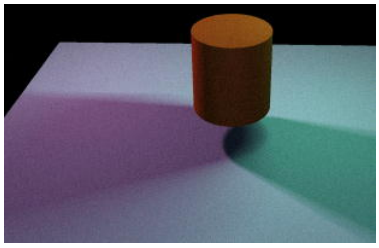
7. Adjust the Quality value to about 10. The rule of thumb for this quality setting is the same as for the point light source: the bigger/closer the light source, the higher quality settings will be required.

8. Render the view to see the result. All shadows have smooth edges. Note also that the color of sub objects of the special light defines the color of the light.



Note

Sub objects of a special light source do not define any visible surface - use object duplicates for this purpose.



Shadows created by a special light source

Volumetric Rendering

Volumetric rendering allows you to visualize fog, smoke, fire and other atmospheric effects. For example, a light source can illuminate a fog cloud. Objects can cast shadows through a fog cloud, just like in the real world.

Defining such effects is simple and logical: first a suitable volumetric object shape is created, then a suitable material is applied to it. The program takes care of the rest. In other words, properties of foggy and gaseous objects are defined and controlled exactly the same way as properties of solid objects: using materials.

The Materials chapter explains how to build fog-like materials. This section demonstrates how you can use them in your scenes.

Defining volumetric properties of geometric objects

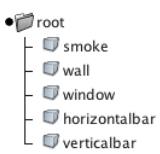
In the first example we will create a dark room filled with smoke. Light shines through a window, illuminating the smoke and casting shadows through it.

Tutorial level: Medium

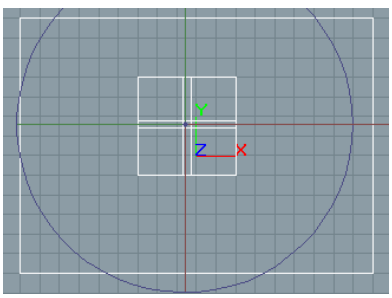
Sample project: 'tutorprojects\rendering\smokyroom'

1. The first task is to model the room, or actually only one wall with a window, which is enough to show the effect. Take a front view and turn grid drawing and snap to grid on.

2. Go to the Analytic tab and select the analytic cube tool. Draw a large cube filling most of the view window. Using the select window's Rename tool (you can find it from Geometric Objects tab popup menu), rename the cube as 'smoke'.



The components of the room



3. Create 4 more analytical cubes as shown in the example image. The first new cube should have the same size as the smoke cube, because it will be the back wall behind the smoke. The second cube is the window in the middle of the wall. The third and fourth cubes are the window bars. Rename the cubes as you create them to make identification easier.

4. Take a side view. All the cubes we just created overlap each other and their thickness is not correct. Select smoke, and using either the Extend tool or the scale handles, increase the thickness of the smoke so that it fills at least half of the view window. Then move it to the right side of the other cubes either using the move handle or the Move tool.



Note

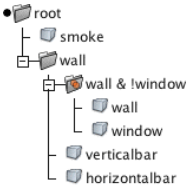
Leave a thin gap between the smoke and the other cubes. Always avoid situations where two surfacEs (the smoke and the wall in this case) overlap exactly. Otherwise the rendering engine gets inconsistent results in computing where the fog ends and the wall starts. Remember that computers have finite accuracy. The gap can be very small, only a fraction of one millimeter. You may have to disable grid snapping temporarily to be able to position the smoke at a small distance from the wall.

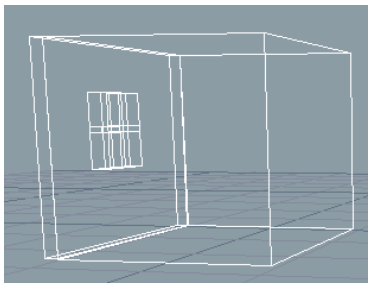
5. Then select wall, window and the two window bars. Move them into position, if necessary. Use the extend tool (or the scale handle) to make them somewhat thinner. Remember to maintain a suitable thin gap to the smoke.

6. Select the window and increase its thickness slightly, so that it sticks out from the wall. We will use it for cutting a hole to the wall, and it should cut all the way through.

7. Select wall, then holding the Shift key down, add window to the selection. Then activate the Boolean tool from the Boolean tab of the toolbar. Click on the Remove icon, then click Accept. Now the window cuts a hole in the wall.

8. The default wireframe representation of the window is quite bad. If you like, you can improve it in the following way: Select the wall & !window level. Then open the hierarchy of the level, select the original wall and window objects and clear Invisible in real time rendering property from the Gen tab. Close the property window.





Room finished. There is a small gap between the smoke cube and the back wall.

9. Let's clean up the hierarchy a bit. Multi select the wall & !window level and the two window bars. Then apply Drop to a Level from the popup menu. Rename the created level as 'wall'. The structure of the scene is now almost ready.

9. Select the smoke object. Go to the Materials tab of the Select window and drag & drop the shadowfog material onto the view window.

10. Take a side view and zoom out. Go to the Light Sources tab and activate, for example, the Point tool. Click behind the room so that the light will shine through the window to the room (don't put the light above the room, because there is no ceiling). Click towards the room a second time to finish the light source. Rename it to 'Sun'.

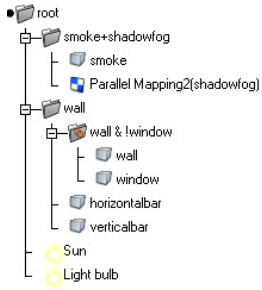
11. Make sure that the view uses reasonable quality or quality over speed rendering settings. If not, drop either of them or another suitable rendering settings object to the view. Take a perspective view. Zoom and rotate the view so that you can see the interior of the room in a suitable way. Then click the Render button.

12. It may happen that you do not see the fog, or the fog looks much too bright. The reason is that shading of most fog like materials strongly depends on the scale of the model. A big cloud in the sky can be several kilometers thick and can hide a bright object like the sun. A smoke from a candle is a couple of centimeters thick and hardly visible at all. To change the strength of the smoke shading, select the shadowfog material in the Material tab of the select window, and open the property window. Increase the Turbidity property to make the fog thicker. You may also change the Color of the fog to light blue.

13. The accuracy of the shadow appearing in the smoke can be improved by increasing the Sampling value in the property window of the material. This may be necessary if the object casting the shadow has small details (such as thin window bars). For example, set the Sampling to 20. The higher the value the better the quality (and the slower the rendering). Advanced users may add random variation to the sampling positions using

VSL (see 'additive fog' VSL example for details). Together with adaptive anti-aliasing, this method produces accurate high quality shadows with a reasonably low sampling rate.

14. If the smoke still appears too weak, you can increase the brightness of the light source. Select the sun light and use the Intensity slider in the property window's Spec tab. Try values ranging from 1 to 2. You may also create a new, weaker light source on the top of the room to simulate a light bulb in the ceiling.



Smoky room finished

The example project 'smokyroom' uses the randomized sampling method mentioned above for improving shadow accuracy. You can re-use the material in other scenes. To render a high quality still image, select Stochastic anti-aliasing at level 2 or higher and set anti-aliasing threshold to 5 or lower. This ensures that fog effect computations are made very accurately on areas where fog illumination changes rapidly.



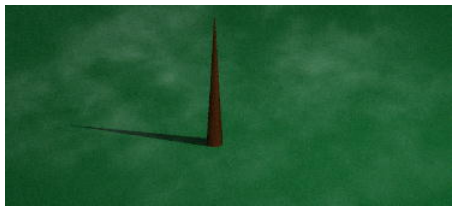
Fog clouds

Tutorial level: Medium

Sample projects: 'tutorprojects\rendering\fogcloud' and 'tutorprojects\rendering\irregularcloud'

You can simulate a fog cloud the same way as smoke in the previous example. Often a fog cloud is not homogeneous: there are dense areas and clear areas quite randomly. This effect can be achieved in two ways:

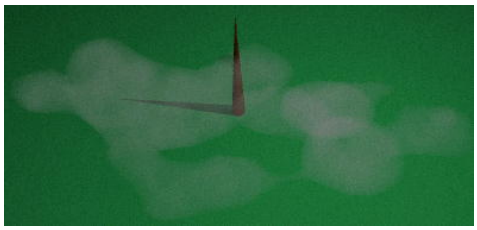
1. Use one big object (e.g. a flat ellipsoid) to model the fog cloud and apply a material, which defines a varying density. The density function must be carefully designed to hide the symmetric shape of the ellipsoid. The adjacent image shows an example.



A simple elliptic fog cloud. Irregularity in fog density hides the simplicity of the shape

2. Model an irregular shape using metaballs or subdivision surfaces. Remember to set high enough recursion depth under the renderings settings tab (10 in the example image).

The fog material can be self-illuminating ('plasma') or diffuse-illuminated ('shadowfog'), or the combination of the two.



A fog cloud consisting of metaballs

A note for users of previous Realsoft products: In Real 3D, overlapping areas of transparent objects, which shared the same material, were automatically melted together. This does not happen in Realsoft 3D. For example, using a large set of analytical spheres for representing an irregular fog cloud does not produce a good result, because the thin seams between spheres usually affect the shading. Use the metaball object for this kind of applications.

Global atmospheric effects

Global atmospheric effects, such as a global fog, can be created easily with the Creation/Atmosphere tool. This tool creates an object, which defines volumetric properties for the entire (infinite) space.

Tutorial level: Beginner

Sample project: 'tutorprojects\rendering\globalfog'

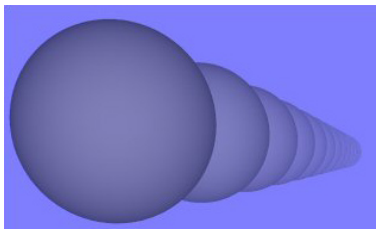
To create a global fog:

1. Activate the Atmosphere tool from the Light sources toolbar tab.

2. Select a fog material from the material list in the tool control bar.

3. Click Accept. You have now defined a global fog.

4. Create a couple of spheres at different distances and render. The higher the distance, the better the fog obscures the spheres.



The Atmosphere tool and a sample scene rendered



Note

Use only self-illuminating, homogeneous fog materials with the global fog. The shading of other kind of fogs is performed with a finite amount of samples, which are not suited to the infinite nature of the global fog object.

Visible lights

The spotlight tool allows you to select a volumetric material, which will visualize the light beams.

Tutorial level: Beginner/Medium

Sample project: 'tutorprojects\material\visiblespot'



Spot Light tool

1. Go to the Light Sources tab of the toolbar and activate the Spot Light tool.

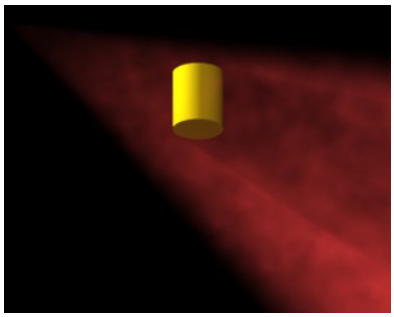
2. Select one of the materials available in the material list of the tool control bar. The list shows only those materials of the current library, which define volumetric effects. If the list is empty, all such materials have been deleted, and you should reload the standard material library. Select shadowfog.

3. Define the spot light geometry as usual.

4. Place an object inside the light cone so that it will cast a shadow.

5. Render. If the light beam is not visible enough, double click the shadowfog material in the Material tab of the select window and double the Turbidity value in the opened property window. Test render and increase Turbidity again until a suitable effect is achieved.

6. The sample image shows growing brightness as the thickness of the beam increases. This can be changed by selecting Falloff = Distance or Falloff = Distance squared from the spotlight's Spec controls on the property window.



Fog makes spotlight's beam visible

If you now look at the select window's object geometry tab, there are no material mapping objects at all. The reason is that a spotlight can have a material as a direct attribute. If you open the property window and go to the spot light's Spec properties, you can see that there is a Material gadget available. From the gadget you can change the material or disable it by selecting the empty entry of the list.

In the sample project, 'tutorprojects\material\visiblespot', an additive fog material, which is described in the 'Fog Examples' section of the VSL tutorials, is applied to a spotlight. Because of its additive nature, the illuminated fog never darkens objects behind it and is therefore especially suitable for lighting simulation.

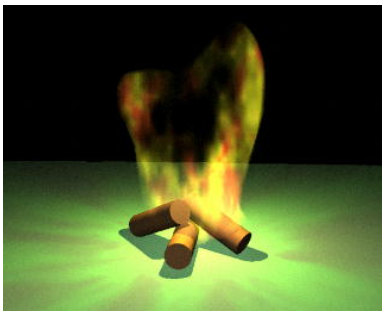
Volumetric fire

Fire can be simulated using suitable materials. Let's experiment using the Fire material in the default material library.

Level: Medium

Sample project: 'tutorprojects\rendering\simplefire'

1. Create a green analytic rectangle defining the ground level and rename it to 'Ground'. Create a pile of brown analytic cylinders representing firewood above the ground. Drop the cylinders to a level and rename the level to 'Firewood'.



Volumetric fire material

2. We could use a simple elliptic sphere to define the volume of the sphere. However, a more interesting result can be obtained using metaballs. Go to the geometric objects tab of the select window and select New/Metaball from the popup menu.

3. Double click the created metaball object in the select window to open the property window. Go to the Spec tab and set each of the Resolution X, Y and Z sliders to a value between 10 and 15. The surface of the metaball structure does not have to be smooth because only volumetric shading will be applied to it. Decreasing resolution will improve rendering speed. Set Field to Quadric (smooth blending of metaballs) and Iso Value to 0.4 (to fill most of the volume of metaballs). Activate the Simulation checkbox on the top part of the Spec tab. We will animate the metaballs using a gravity field later.

4. Create an analytic sphere above the firewood. The size of the sphere may be one quarter of the size of the total fire. Create 3-6 more spheres around it at random positions. The spheres should partially overlap. You may stretch some or all of the spheres vertically higher so that the shape looks more like a flame.

5. Select the first sphere. Open the property window, go to the Sim tab and set Gravity to Cause. The center sphere will create a gravity field in which the other spheres pulsate. Switch to The Phys tab and set mass to 100 000 000 (number 1 and eight zeros = one hundred million) kilograms. A gigantic mass like this is necessary, because Realsoft 3D uses real world units in simulation. Two apples on a table do not attract each other much, do they? Only a planet-scale mass is sufficient to create a noticeable gravity field.

6. Select all the other spheres. Their default masses are OK, but go to the Sim tab and set Gravity to Affected. These lightweight spheres will bounce around in the gravity field caused by The heavy center sphere.

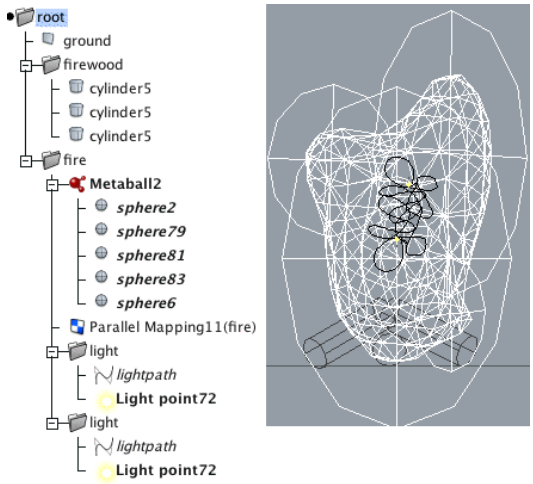
7. Select all the spheres, and drag & drop them to the metaball level on the select window. An irregular meta surface appears. Edit the spheres inside the metaball level if the shape is not good enough. After that, if you like, you may set the actual spheres Invisible in real time rendering in the property window Gen tab, to clean up the wireframes.

8. Select the metaball object. Drag the material Fire from the Select window's Materials tab to the view window. This creates a new level, rename the level to 'fire'.

9. Increase Recursion depth of the rendering settings used to at least 6 and test render to see how the fire looks. If the brightness of the fire is too low, open the property window and increase the Flame saturation value of the fire material. The Flame density attribute defines how big part of the fire volume becomes filled with flames. Too high a value fills it completely, and reveals the original shape of the fire volume clearly. This is undesirable if the volume is relatively simple. You may also adjust the size of the fire parallel map to change the size of flames appearing in the material.

10. The fire is supposed to cast light to its environment. Therefore, create two point light sources inside the fire. Change the color of light sources to a suitable light yellow. One of the light sources can be non-shadow casting (property window's Spec tab/Shadow Casting set to No shadows) to save some rendering time. Create

two closed nurbs curves traveling around in the fire (see the figure below). Multi select the first light source and the first nurbs curve (in this order). Then select Animation/Path/Move from the main menu bar. Then drop the light source and the curve to a level. Do the same for the second light source and curve. The light sources follow the paths, casting changing shadows around (If set back on).



The final structure of the fire scene

The simple fire scene is now ready. Play the animation to see animated fire. If the fluctuation of the metaball shape is too slow, increase the mass of the center sphere. Decrease the mass if things happen too rapidly. A ray traced preview rendering to an AVI file shows another animation effect: the material defined flames also change and travel upwards. You can control the speed of this effect using the Flaming speed attribute of the fire material.

The workflow above shows only the basic ideas for a simple fire. The realism can be improved in many ways: adding new material details, more metaballs, etc. In the 'simplefire' sample project the Volume properties shader was duplicated, changed to a Surface filtering shader and edited slightly. The new shader filters the color of the light sources inside the fire with the color of the flames. This makes illumination of the environment much more interesting and realistic.

Rendering settings and volumetric effects

Most of the default rendering settings are suitable for rendering volumetric effects. Only the draft quality settings (speed over quality) are too simple for this purpose. The general guidelines are:

- In order to render volumetric effects, the recursion depth must be two or higher. Recursion level three is usually the minimal sensible value. Complicated fog clouds (defined by metaballs, etc.) may require a recursion depth from 5 to 10.
- You can keep all the check boxes in the Detect frame activated. Because of internal optimizations, they don't slow down rendering when you haven't defined volumetric effects. You need to modify these options only if you have included volumetric effects in the scene and, for whatever reason, want to disable them.
- If the scene includes scanline particles, check the Visible in Reflections/Refractions/Fog option of the scanline rendering settings.

Motion Blur

The exposure of a film camera takes a certain time period. Non-static objects move and change their appearance during the exposure time. Because the film records everything that happens during the exposure time, such objects do not appear sharp but blurred in the picture. This phenomenon is called motion blur and can be simulated in Realsoft 3D.

Motion blur can be used as an effect to increase the realism of images. It also improves the quality of animations. Common video and film standards show 20-30 frames per second. At such frame rates and without motion blur, the animation of a rapidly moving object may not appear continuous. Various strobing artifacts may also become visible.

The effect of motion blur is visible (and necessary) only when the animation contains objects moving a certain amount per frame. As a general rule, an object should move at least 10 pixels between two frames before

the blur produces noticeable results. The use of motion blur makes rendering slower. For example, 9 motion samples usually at least triple the rendering time.

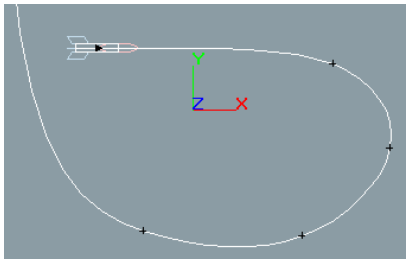
The motion blur feature of Realsoft 3D works by simulating physical reality accurately. Moving objects, shadows of moving objects, animated textures and materials become blurred if they change during the exposure of a single frame. If the camera moves itself, the whole world becomes blurred.

Rendering a motion blurred animation

When building a scene, you do not have to worry about motion blur. If you decide to use motion blur after the animated scene is finished, all you have to do is to set some options in the animation and rendering settings. The following example demonstrates this:

Tutorial level: Medium

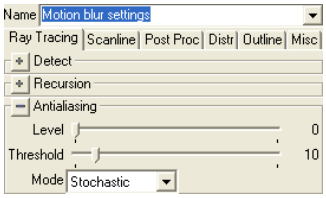
Example project: 'tutorprojects\rendering\missile'



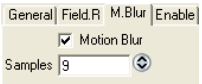
A missile follows a path

1. Create the animation in the usual way. For example, model a missile and draw a path (a NURBS curve) for it. Multi select the missile and the path and apply Animation/Path/Directed from the main menu.

2. Go to the Rendering settings section of the select window. Select the settings you plan to use for rendering. Open the property window from the select window's popup menu and go to the Ray Tracing tab. The rendering settings can be as usual, but set the anti-aliasing Mode to Stochastic and Level to zero. The motion blur system will take several samples per pixel over time, and therefore the usual anti-aliasing is not necessary. Only when using a minimal amount of motion samples, e.g. 4 samples between each frames, you should define some anti-aliasing, for example level 2. This step is important because it will save a lot of time in rendering the animation. Close the property window.



Defining anti-aliasing settings for motion blur



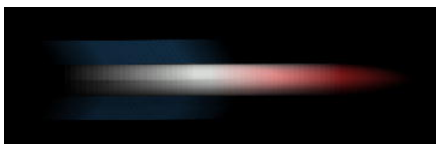
Activate motion blur and select the amount of motion samples

3. Open the Animation Settings window from the Windows menu of the main menu bar. Check the Motion Blur option and set Samples to 9. Nine motion samples are enough for most purposes, and does not slow the rendering too much. Close the window.

5. Let's preview the motion blur on the view window first. Drag and drop the rendering settings object, which was prepared in step 2, to the view window (this is not necessary if you know that the view already uses the correct settings). Open the view property window, go to the Render tab and activate the Continuous rendering option. Close the view property window.

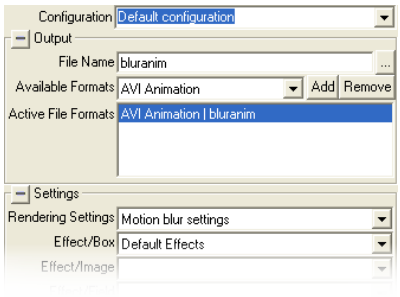
6. Open the animation window from the View main menu (if it is not already open). Hit the Play forward button. The program starts playing the animation, and renders the view after each time change because of the enabled continuous rendering option. The first rendered images look quite bad, because anti-aliasing is disabled.

Wait until nine images have been rendered. At that point, the program computes and displays a motion-blurred image in the view. You may stop the animation playback now. Open the view property window and turn off Continuous rendering option. Then rewind the animation. If the blur quality was not good enough (the missile moves too much between the motion blur samples) adjust the amount of samples to 16 using the animation settings window.



Moving missile appears blurred

7. The final step is rendering to a file. Select File/Render from the main menu. Prepare file rendering the usual way, like instructed in the manual section 'Rendering Still Images and Animations to a File', but remember to select the correct Rendering settings. Then start file rendering by hitting the Play button.



Rendering settings for motion blurred file rendering include stochastic 0-level anti-aliasing

Field Rendering

When very smooth animation is required, 25 frames per second (or even 30 fps) may not be enough. Extra smoothness of motions can be achieved by showing a new image in each video field. The PAL system, for example, shows 50 individual fields per second. Nevertheless, because of the interlaced nature of the PAL system, each field includes only one half of the vertical picture lines.

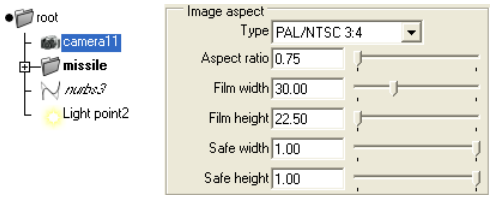
Therefore, rendering full video resolution images for field-rate recording means wasting one half of the rendering time and picture information.

Rendering fields to disk files

Using field rendering in Realsoft 3D is easy: build and animate the scene in the normal way. Then activate Field rendering from the animation options before starting the rendering. The example below shows how to prepare field rendering of an animation for PAL video recording.

1. Create the animation as usual. Use the normal amount of frames (25 frames for one second in PAL).

2. Insert a camera object to the scene using the camera tool, if not already inserted. Open the property window and go to the Spec tab. Set the Image aspect to PAL from the Presets list. Close the property window.



Selecting PAL camera aspect ratio



3. Open the Animation Settings window from the Windows menu of the main menu bar. Check the Field Rendering option.

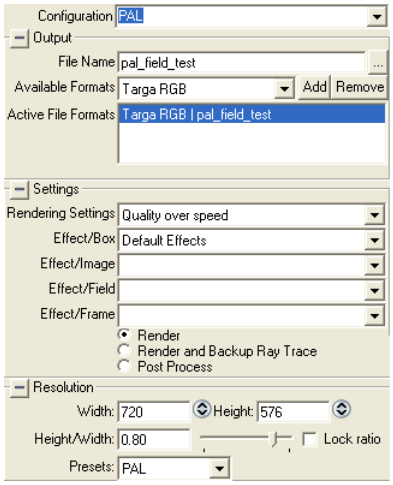
4. The animation settings window has also two other options for controlling field rendering. The options Shift latter field up and Interlace latter field above should be cleared if your video hardware defaults to the EVEN field first. If the hardware uses the ODD field first, set both check boxes.

5. The final step is rendering to a file. Select File/Render from the main menu. Select PAL from the configuration list. It already includes the correct image resolution.



Note

The resolution is the frame resolution, not the field resolution. Prepare file rendering the usual way, as instructed in the manual section 'Rendering Still Images and Animations to a File'. Then start file rendering.



File rendering options for PAL fields. The selected render settings object has appropriate field options.

Scanline Hairs

Scanline particles are ideally suited to small and thin objects such as hairs. The following tutorial demonstrates a typical workflow for creating hair:



A haircut modeled using 2D particles

Tutorial level: Advanced

Example project: 'tutorprojects/rendering/hairs'

1. Load in the object 'models/nurbs/simple head' (or model the head yourself).



Options for Uvimage tool

2. We start by texturing the hairy area of the head with a material that looks like hairs. This helps to keep the amount of real hairs reasonably low. Select the loaded head object. Go to the Materials tab of the toolbar and activate the UVimage tool. Define a suitable file name for storing the image, set color to black and press Accept.

3. Turn shaded OpenGL drawing on from the compass menu. Because of the attached black uvimage, the head appears black. Paint the area, which should be covered with hair, using the 3D painting tools (included in the Materials tool tab). Use the tools as follows:

- The paint color should be white. The image will be used later as a scope map, and white means fully hairy material.
- Paint the border of the hairy area with the airbrush tool to achieve a smooth gradient. You can reduce the hairy area by carefully brushing some black color.
- You can fill the hairy area using the Curve painting tool with the filled option set.
- It may happen that some painting appears on the neck when you paint the top of the head. Don't worry, this happens because the default UV space is not properly initialized. You can afterwards assign pointwise UV coordinates to the Nurbs model using the Define UV tool, which appears to the tool control bar when the head is selected. Then open the UV Editor window, set the head to edit mode, select the lowest point row from the view window and move it slightly downwards in the UV editor.



The hairy area painted white

4. After painting, select the head, open the property window and go to the Col tab. Clear the Texture field, which contains the name of the painted image, and hit Enter. Set the Surface Properties/Color gadget to a suitable skin color (e.g. light brown 1 0.8 0.7).

Materials for the hairy skin

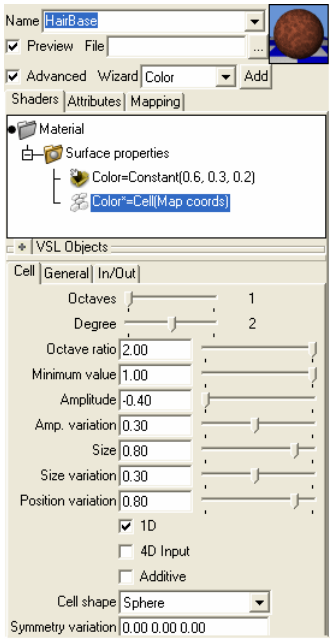
5. Next we define the materials for the hairy skin. Three materials are needed for this purpose:

- A material for hairy skin
- A scope map material which defines which part of the head is hairy
- A level material combining the two above mentioned materials

Change the select window to show the material library. Select twice New/ VSL Material and once New/Level Material from the select window's popup menu.

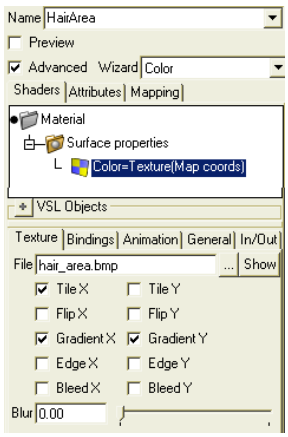
6. Select the first created VSL material. Rename it as hairbase. Enable Advanced options of the property window and add the following VSL objects to it:

- A Surface properties shader
- A Constant object that assigns a suitable hair color (e.g. dark brown 0.6 0.3 0.2)
- A Cell object, which adds irregular dark spots to the surface by scaling the color. Set Operation to Multiply from the popup menu. Minimum value should be 1 and Amplitude -0.4. This gives a range from 0.6 to 1 (in other words, darkest spots are 40 % darker than the base color). All Variation fields should be set to a non-zero value, so that the pattern becomes irregular. See the picture. Check the 1D option.

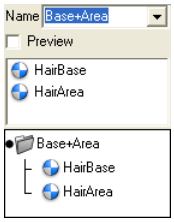


The base color of hairy skin is dark brown. Cell noise adds small darker spots to it.

7. Select the second blank VSL material. Rename it as HairArea. Add a Surface properties shader and a Texture object to it. Set the output channel of the Texture object to Surface:Scope and input to Surface:UV coords. Set the File name gadget to the name of the image, which was initialized in the step 2 above and painted later. Other options are shown in the example image.



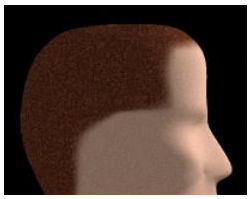
A scope map material for the hairy area



The level material combines HairBase and HairArea to one single material

8. Select the Level type material and rename it as Base +Area. Using the property window, drag & drop the materials HairBase and HairArea to the sub material list.

9. Having the level material (and the head object) still active, select Mapping/Parallel from the select window's popup menu. Hold down the shift key and draw a small parallel mapping cube over the head. The smaller the mapping, the denser the pattern. Test render the view window and adjust mapping size until the result is suitable.

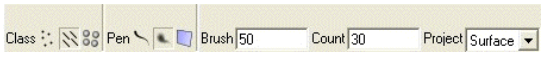


A pattern of tiny dots mapped to the head

Adding the hairs

10. Now we can start adding the actual hairs. Switch to the Geometric Objects tab of the Select window and select the original head object (not the level containing it). Go to the Wire tab of the property window and set Texture Quality to for example 128. Wait for a while until the HairBase texture pattern appears to the view window. Seeing the hairy area, which was already 3d painted to the head, will guide us where to add the hairs.

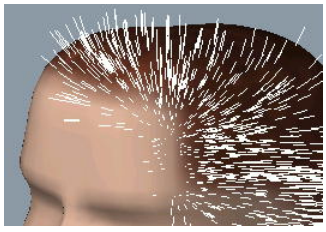
11. Zoom in so that the head fills most of the view. Activate the Particle tool from the Creation tab of the toolbar. Select airbrush painting, 2D particles, Line Width = 50, Count = 30 and Projection = Surface from the tool control bar.



Hairbrush options

12. Start painting the hair over the head by dragging with the left mouse button over the hairy areas. Rotate the view occasionally so that all areas get painted. Do not worry if hairs are added to the wrong place such as the nose - they can be removed later. On the contrary, brush a fair amount of hairs over the boundary of the hairy area, so that the boundary will have enough hairs. Click Accept when the hair is reasonably dense everywhere.

A suitable amount of hairs is 5000 -10 000. Using a very high amount can make screen refreshing slow and cause memory problems later in ray trace rendering. If this happens, it is good to know that using a high box count in box rendering decreases the memory requirements. You can check the amount of hairs from the property window's spec tab after creating the particle object.



Airbrushing hair particles onto the head. Note how hairs overlap the actual hair boundary at this stage.



Note

The tool inserts 2d particles perpendicular to the surface. It may happen that the surface normal points to the opposite direction from what was intended for hairs. If this happens, cancel the tool, take a top view and mirror the head. Mirroring inverts the normals. Then start the particle tool again.



Lasso selecting the hairs that are outside the hairline for deletion

13. Take a side view and set the created 2D particle object to its Edit state using the view's popup menu. Press the 4 key down and start dragging the lasso selector along the hair boundary. Draw a loop that includes all hairs that lie outside the hair-colored part of the head.

14. If the haircut is symmetrical, you can now click the Delete tool of the tool control bar. Otherwise take a top view and Alt-drag the hairs of the opposite side to remove them from the selection before applying Delete. Then repeat the selection and deletion from the opposite side.

15. Turn off the Edit mode. Make sure the particle object is selected. Go to the property window's Spec tab and select the Geometry sub tab. Pick Length from the long attribute list and set it to 0.01 meters (1 cm is pretty good for this kind of simple haircut). Pick Diameter1 and set it to a small value like 0.001 (1 mm). Pick Diameter2 and set it to zero. You may try different values later but these should produce a reasonably good result.

16. Go to the Col tab and set the color of the hairs to the same color as you used for the hairbase material (0.6 0.3 0.2).

17. Test render and adjust the particle properties as required.

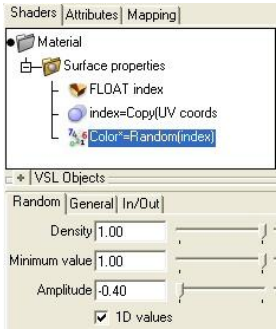
Coloring the hairs

Most hair types have a subtle color variation. When you grow older (or a software product release becomes badly delayed), gray hairs start appearing. Let us add this property to the hair next.

18. Switch to the Materials tab of the Select window. Select New/VSL Material from the popup.

19. Rename the material as HairColor and using the property window, add the following VSL objects to it:

- A Surface properties shader. Properties of scanline particles are normally defined in this shader.
- One Float type variable called index
- A Copy object that copies UV channel's last sub channel to the index variable, set Sub channel 1 = Z.
- A Random object with the following options:
  - Output = Surface:Color (because this material modifies hair color)
  - Input = index. The random variation is computed from UV channel's third component, which is the index of the hair.
  - Operation = Multiply (random factor scales the original tone)
  - Density = 1 (every hair has a slightly different random color)
  - Minimum value = 1, Amplitude = -0.4 (range 0.6 …1 makes darkest hairs 40 % darker than lightest hairs)
  - check 1D values



A material that randomizes color of hairs

20. Drag & drop the created material to the view window (hair particles should be still selected). A new test rendering finishes this tutorial.

Tips and tricks

- Curved and curly hairs can be modeled using Nurbs curves. Clear the Invisible in photorealistic rendering flag to enable scanline rendering. The Spec tab contains thickness controls for both ends of a curve. Lattice/Distribute and Lattice/Paste on Surface tools can be used for attaching curves into surfaces.
- The hair color material can also include a shader defining some specular shine. The sample material tutorprojects/rendering/shinyhair contains an example material. It uses the Surface:U direction channel, which contains the direction of hairs.



Shiny black hairs

Alpha Channel

Many video compositing related projects require use of an additional 8 bit mask channel, usually called as Alpha channel, which controls how rendered graphics is inserted over other background graphics. This tutorial introduces the basics of Alpha channel use in Realsoft 3D.

Tutorial level: Medium

Example project: tutorprojects/rendering/alpha_ready

1. Model a simple test scene of a magnifying glass floating in space or load the provided test scene 'tutorprojects/rendering/alpha_scene'.

2. We will render the alpha channel information to a disk file using a format which supports the extra channel. Therefore, go to the 5th tab of the select window; it contains the output object library. If the list contains a Targa object, LMB click it to select it. If not, apply the popup menu New/Targa. Open the property window and make sure that the Alpha option is set. You may also set the Compress option to save some disk space.

3. Open the file render window from the pull down menu File/ Render. Add the Targa output to the Active File Formats list and remove other active outputs. Define an appropriate file name, for example 'my_targa_test'. Then select for example rendering settings Quality_Over_Speed. Resolution can be 600*450 pixels. Hit the render button and wait until the image is ready.

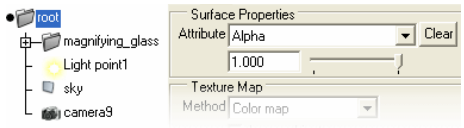


The test scene rendered

4. Start the Show Image utility program; you can find it from the Realsoft program group of the Windows Start Menu. Use its File/Open pull down menu to select the rendered Targa file. If you used the example scene, the picture will look like in the adjacent example image.

Now use the Channels pull down menu of Show Image to view the Alpha channel. It is completely black! This is because we did not specify any Alpha values for objects. By default, the program processes no Alpha channel information at all, in order to speed up rendering in projects which do not use such information. Fortunately, it is very easy to assign the required Alpha values to objects, as we will see next.

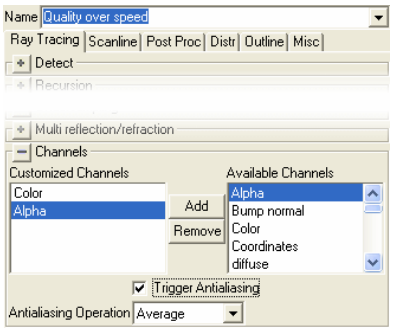
5. Switch back to Realsoft 3D and select the root level of the hierarchy of geometric objects. Open the property window and go to the second Col tab. Select Alpha from the Attribute list and move the slider below to the full value 1.0. Now all objects have this Alpha value, because the target was the root level. Only the empty background space renders with zero Alpha.



Setting Alpha value for the whole scene

The root level defined alpha can be overruled in deeper hierarchy as follows: open the hierarchy of the magnifying glass object, select the item 'lens' and change its Alpha to a semi transparent value 0.5.

6. Let's tailor the antialiasing settings for optimal alpha channel quality. Switch to the Rendering Settings tab of the select window. On the property window, select the leftmost tab and open the Channels frame on the bottom. Pick Alpha from the Available Channels list. Hit the Add button so that Alpha appears into the Customized Channels list. Activate the inserted item and turn the Trigger Antialiasing option on. Now, not only color changes but also Alpha channel variations trigger improved sampling at object boundaries. This ensures good antialiasing for the Alpha channel.



Alpha channel Antialiasing controls

7. Go again to the File Render window and restart rendering. Then view the new image version in Show Image. Choose Alpha from the Channels menu to find out that now the image contains a nicely antialiased Alpha component.

More advanced information about Alpha channel use is included in the Compositing tutorial.



The Alpha channel of the final Targa image

Advanced Illumination Examples

Illumination Maps

Big part of real-world illumination is created by surfaces that reflect illumination from actual light sources. White walls of a room spread the light from the lamp in the ceiling to shadowed places such as behind the furniture.

The UVImage tool can evaluate and store this kind of indirect lighting to so called illumination maps. The following simple example shows how to use this feature.

1. Model a simple scene shown in the image on the right: a rectangular floor surface, one wall on the left and a table standing on the floor. Alternatively, you can load the sample scene 'tutorprojects/rendering/table'. The table consists of a cube and four cylinders. Place one point light source on top and to the right of the table. Open the property window of the light source and adjust its Intensity to level 1.0. You may also open the view property window and set Camera tab's Flashlight to black color (0,0,0). Then the only light source of the scene is the point light and shadows will be quite dramatic.



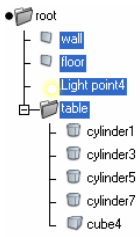
A table illuminated by a single light source

2. Next we generate the illumination maps. Go to the Materials tab of the toolbar and activate the UVImage tool. The default resolution is fine, but change the Type selector to Indirect lighting. If you have time to wait a bit more accurate computation (and better quality), increase the Samples value from 1 to 3 - 5.



UVImage options for indirect illumination

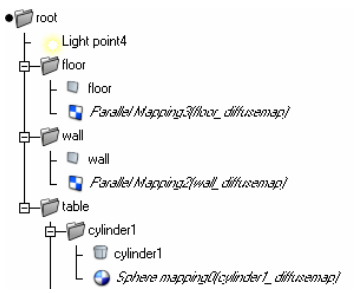
3. Select all objects for which you want to generate maps - in this case the wall, the floor and the table. Select also the point light source. This is important; surfaces have always two sides and the program must know for which side the illumination should be evaluated. The selected light source indicates it. Click Accept.



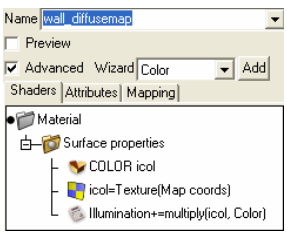
Object selection for illumination evaluation

Wait until the progress indicator shows that the tool has finished. Now look at the select window: the object hierarchy has changed slightly because the tool has inserted new illumination maps to the hierarchy. The material library contains a set of new materials as well.

4. Hit the render button to ray trace an image of the scene. The space under the table, which was previously totally dark, now has some subtle illumination. The indirect illumination is usually weak compared to direct lighting. If you wish to enhance it afterwards, you can simply reduce the intensity of the point light source to one half and double the film sensitivity of the view camera from the view property window.



The hierarchy with the illumination maps



The diffuse illumination material

You can naturally edit the diffuse illumination materials if you need to control illumination of a single object.

The tutorial example shows that illumination mapping is not part of the rendering process but a tool that changes the scene. This means that it is best suited for situations where it is necessary to make illumination of a static scene more realistic. The time consuming illumination maps are generated only once; after that, the camera can be animated to move in the scene. One can also add some animated objects whose contribution is not significant for indirect illumination.



The table scene with indirect illumination maps



Note

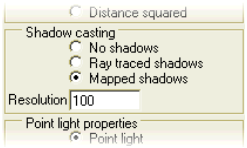
Before applying illumination maps to SDS objects, remember to initialize proper UV coordinates for them first.

Mapped Shadows

Shadow mapping is another texture mapping based illumination method. Mapped shadows usually have nice smooth edges which increase realism without the high rendering time penalty of ray traced smooth shadows. On the other hand, shadow mapping often suffers from a number of accuracy problems. It may take several rendering attempts to adjust them until the desired result is achieved.

Let us continue experimenting with the table scene of the previous chapter. The indirect illumination maps may be included, but you can also reload the table scene 'tutorprojects/rendering/table'.

1. Select the point light source and open the property window. Go to the Spec tab. Then change the Shadow casting selector from Ray traced shadows to Mapped shadows. Then render the view window to see the result.

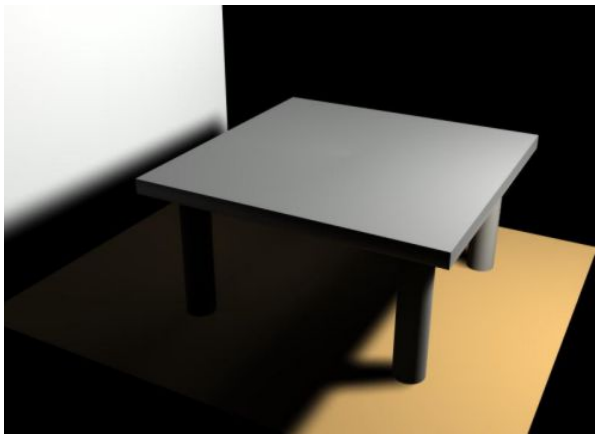


Selecting mapped shadows

Although shadows now appear smooth-edged, some quality problems can be detected easily. The shadow on the side wall has some banding, and the top part of the legs become illuminated through the table top! The default shadow map resolution is most likely not enough.

2. Make sure that the light source is select and go back to the property window. The Spec tab shows that the default Shadow map resolution is 100 pixels (actually 100*100 pixels, the resolution is always the same in both dimensions). Change the value to 400 pixels and render again. Now the quality is much better.

3. We finish this example by examining how to control smoothness of mapped shadows. Take a look at the property window of the light source again. The bottom part of the window contains a Size field, which by default has the value of 0.1 meters. Double the value to 0.2 meters and render the view to see how the shadows change. The bigger the light source, the smoother the shadows. It is usually necessary to keep the size in a sensible value which matches the scene dimensions. A distant light source hundreds of meters away from the target should have a diameter of several meters to function properly.



Mapped shadows have smooth edges

Room Illumination

The next example demonstrates the same illumination techniques as the chapter above: indirect illumination maps and mapped shadows. This time the scene is a bit more complicated, a room interior shown below. The picture shows how the example scene 'tutorprojects/rendering/room' looks like when rendered with two point light sources (lamp, sun) and some flashlight.



A room illuminated by two point light sources

First we generate the illumination maps:

1. Select the point light source inside the lamp object. Then multiselect the room level. We will generate the maps for only the largest surfaces of the scene, not for the furniture or other details.

2. Activate the UVImage tool from the Materials tooltab. Change Type to Indirect Lighting, set Samples value to 3-5 and hit OK.

3. Wait until the UVImage tool has finished the map computation. Then select the spot light source behind the room. We will change its shadow type to mapped shadows: open the property window, go to the Spec tab and activate the Mapped shadows option. Increase the resolution to 400 pixels.

4. Since the spot light is far away from the room, increase its size to 1 meter. Large sizes will also make lighting appear smooth: the spotlight will simulate indirect illumination through a window on a bright day rather than direct sunlight.

5. When using mapped shadows with a spotlight, it is best to use a smooth falloff for the spot edge (sharp edge with otherwise smooth shadows looks peculiar). Therefore, increase the Angle2 value to 5 degrees greater than angle1.

6. Now select the light source in the lamp object and activate mapped shadows for it as well. Resolution can be again 200-400 pixels. The default size of 10 centimeters is quite realistic and suitable, so you do not have to change it.

7. Now comes an important step: There are some glass objects in the scene. Mapped shadows can take only the closest (transparent, translucent or non-transparent) surface into account. Therefore, select the sphere which surrounds the lamp light source and make it Shadow Invisible using the Gen tab options of the property window. Do the same for the glass window object as well.

8. Render a new image to see the changes in the illumination. Adjust shadow map resolutions and light source sizes if necessary.



A smoothly illuminated room interior



Note

Shadow mapping can be applied to spot lights and point lights. Of these two, a spot light works much more efficiently and faster with shadow mapping, so try using that light source type whenever possible.



Note

Analytic rectangle is the most accurate and memory efficient target for indirect illumination mapping. A room wall modeled using a cube requires 4-6 times more memory than a single rectangle (for a comparable quality).

Post Processed Illumination

The table scene tutorial above explained how indirect lighting can be defined using illumination maps. There is another, quite different approach to compute indirect illumination: use of post processing methods.

Example project: tutorprojects/rendering/diffuselighting

Tutorial level: advanced

1. Load first the test scene 'tutorprojects/rendering/table'.

2. Go to the Channels tab of the Select window and create two new color channels called diffuse and unshaded. These channels will carry diffuse illumination and unshaded diffuse color information from the ray tracer to the post processing system.

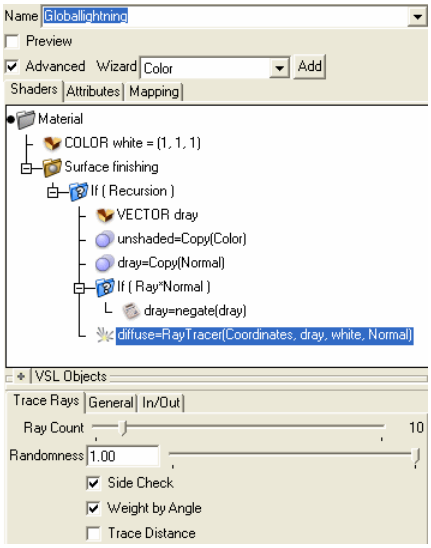


Note

These channels may already exist after loading another illumination project or the default startup file. If they exist, you do not have to create duplicate ones.

3. Switch to the material tab, create a new VSL material 'globallighting'. Check the Advanced option and define the following kind of VSL structure for it:

- The root level of the material contains first a color variable 'white' which is initialized to (1,1,1). The value controls the brightness of indirect illumination.
- The next object in the root level is a shader level of the type Surface finishing. This shader is suitable because it is evaluated only once at the end of surface shading.
- Surface finishing shader contains an If object which examines if Recursion channel is less than 1.01. In other words, diffuse illumination will be examined only at recursion depth 1 (for speed reasons).
- The If block defines one local vector variable called 'dray'. It will hold the surface normal which is turned to point towards the camera.
- Next a Copy object copies the diffuse color of the surface to the unshaded channel.
- Then a Copy object initializes the dray variable from the surface normal.

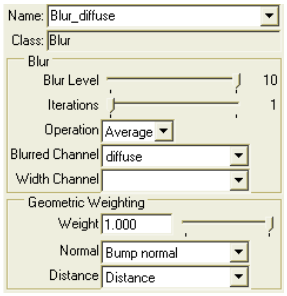


A material that collects illumination information for post processing

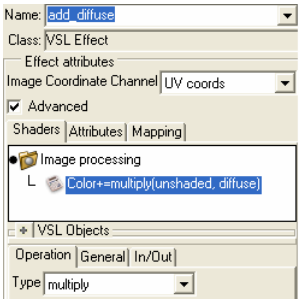
- Next an If shader checks if the normal points away from the camera by examining the dot product ray*normal.
- If the dot product is positive, dray is flipped using an Operation object with Negate. Now dray points to camera's side of the examined surface.
- The final step in the Surface finishing shader is a Raytrace object which traces 10 random rays around the surface normal at camera side of the surface; weighting signal is white because we do not want to premultiply illumination yet.

4. Go to the Geometric Objects tab, select the root level of the scene hierarchy, then switch back to the material library and drag&drop Globallighting material to the view window. The material will shade all objects of the scene.

5. Go to the Post Image Effects tab of the select window. Create a new Blur effect. On the property window, set blur level to a high value 5 - 10. The idea is to average the sampled indirect lighting over the image area. Thanks to this, random variations which would make the image dithered and grainy will disappear. Set Blurred Channel to diffuse (remember that it is the indirect lighting which was sampled by the Raytracer VSL object). Then set Geometric Weighting to level 1. This option controls the averaging so that blurring works only over continuous surface areas - illumination from the floor area that is far behind the table should not affect the table.



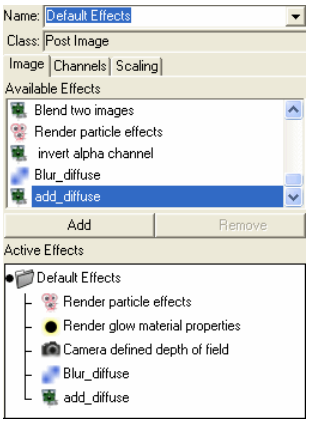
A blur effect averages the diffuse channel



A VSL effect adds indirect illumination to the image

6. Then create a new VSL Effect using the popup menu of the select window. Check the Advanced option. You see an Image processing shader already available. Add one Operation object to the VSL effect. The operation multiplies diffuse with unshaded and adds the result to the Color channel.

7. Select the Default Effects post image or other suitable post processing configuration and add the two new post effects to the end of the Active Effects list - blur first, vsl effect after it.

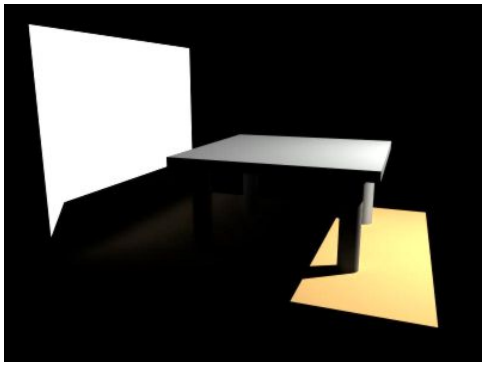


The post processing configuration finished.

8. Drag&drop the Default Effects to the view window. Select a render settings object, for example Reasonable quality, and set Undersampling values to 1. The new diffuse channel is not a trigger channel for undersampling, so it is best to sample every pixel accurately. Drag&drop the modified render settings to the view. Everything is now ready - just render the view to finish the tutorial.

Post processed illumination has some advantages compared to illumination mapping:

- Indirect illumination needs to be evaluated only for visible areas of objects
- Indirect illumination affects all objects of the scene
- There is no need to worry about well defined UV coords of SDS objects
- The technique can be applied to animated objects
- It is possible to ray trace the image once and then post process several times using the backup render features. The brightness of indirect illumination can be easily rescaled in the VSL effect.



Indirect illumination added by the post processing system

There are some disadvantages as well:

- Too low sampling can result to noisy illumination in animations
- Post illumination is computed at every render time, whereas illumination mapping can be computed only once

The globalillumination material above uses its own private recursion depth definition of 1. Indirect illumination is usually much weaker than direct illumination and therefore one recursion level is enough. Anyway, it is easy to apply a higher recursion value. The example project contains a second material version 'GI2' which does that. The second recursion step shading is otherwise similar to step one except that:

- Unshaded signal is tested against a minimal threshold. Sampling happens only if the threshold is exceeded. This both acts as an optimization and keeps indirect illumination shading separated from usual reflections etc.
- The Raytrace object uses much lower sampling value than 10 to keep the render time reasonable.
- This time we do the actual shading instead of only collecting channel information: the result of sampling is premultiplied with the diffuse color and added to illumination channel (the result of the shading will anyway end up to the diffuse channel at the recursion step 1).

Easy Global Illumination

The previous advanced tutorial demonstrated how to configure a global illumination system from scratch. In normal use, the standard components of the installation provide an easy work flow.

Tutorial level: medium

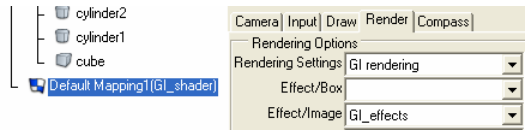
1. The first step is to model a scene as usual. No attention to the GI solution is required at this point. For example, load again the test scene 'tutorprojects/rendering/table'.

2. When the scene is ready (loaded), select the root level of the hierarchy. Then switch to the material tab and drag the material GI_shader onto the view window. This assigns the GI shader to all objects of the scene. If the GI_shader material, for some reason, is missing, you can use New/From Template popup menu to load it from the materials/illumination folder.

3. Next switch to the render settings tab of the select window. Drag and drop the object GI rendering into the view window. This render settings object includes an antialiasing channel setup which is tailored for global illumination. If GI rendering is not included in the settings library, select Paste from file from the popup menu and load it in from the rendersettings disk folder.

4. The next step is to select a suitable post processing configuration. The easiest way is to switch to the Post Image Effects tab and drag & drop the GI_effects post image into the view window. This assigns the effect as an image specific effect. As usual, if the GI_effects effect is missing from the current post effect library, just use Paste from a file popup menu and load in the posteffects/GI_effects.r3d file.

Now the GI setup is ready. Let's continue by adjusting the GI settings to an appropriate level. Since a big part of the GI is computed at post processing time, we can use some time saving tricks here:



The components for global illumination: a root level mapping to a GI shader, a render settings object and a post processing configuration

5. Make sure that Backup Ray Trace option is set in the View/Render popup menu. Then start ray tracing. The program first ray traces an image and finally post processes the global illumination into it.

6. To adjust the brightness of the global illumination into a suitable level, open the property window and select the VSL post effect GI_brightness_control from the select window. The effect has three top level controls. The gamma correction curve controls diffuse indirect illumination. Reflected GI color value scales the brightness of indirect illumination visible via reflections and transparency. GI Contrast adds some contrast to the global illumination. Change the values as you like. Then, instead of rendering everything again, just apply the Post process popup menu of the view window. Now the ray tracing part is skipped and the next image version appears much quicker.

7. The default settings include quite a low amount of indirect light sampling. Consequently, the rendered image may contain some noise which appears as an unwanted brightness variation. To reduce the noise, select the post effect called GI_blur. On the property window, increase the Iterations value by one or two units. Then apply Post process on the view window to see the improved result. If this does not remove the noise, increase the Iterations value further and post process again. A 10 pixel blur repeated 10 times will create 100 pixels wide blur.

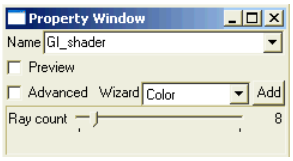
There are other predefined GI components available as well. GI_draft post effect configuration works in a similar way, but it does not anti-aliase the result and is therefore about 4 times faster.

Important notes:

- A wide GI filter may be quite slow to render (several minutes). If the scene is simple and ray traces fast, a quicker way to improve quality is to increase the RayCount1 of the applied GI material. By default, the GI_shader material takes only a couple of illumination samples per surface point. Using for example Ray count value 10 improves quality significantly.
- GI blur filter does complex geometric calculations which require many channels, some of which are in the floating point form. Processing a video resolution image takes several tens of megabytes of RAM. This also means that you should not apply a GI filter as a locally rendered image effect when using a slow internet connection to a render farm. Apply the GI effect per render box.
- If you have originally applied other custom post effects in your scene, remember to insert them also into the global illumination configuration you are using.
- GI filtering may partly remove ray trace time antialiasing. The only reliable solution is to antialiase after post processing i.e. use output down scaling. The GI_effects post effect scales the image by 50 % to ensure a proper quality. GI_draft does not scale the output and is therefore much quicker.

Global Illumination Components

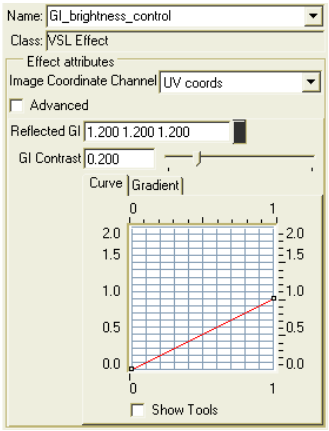
The material which takes care of Global Illumination has only one control, Ray count. It defines how many sampling rays are triggered to examine the indirect illumination of a surface. When these GI sampling rays later hit new surfaces at deeper recursions, only a single ray is launched (for speed reasons).



Usually this is the most important quality setting for indirect illumination. The higher the sampling, the lower Iterations value can be used in the GI_blur filter. The result is finer detail and higher realism in global illumination.

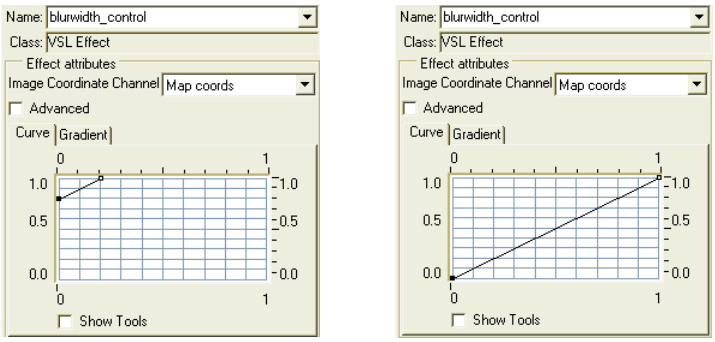
The tracing depth of indirect illumination obeys the general recursion depth of the applied render settings. If you need a dedicated limit for the GI tracing depth, drop the RayTrace VSL object of the GI_shader into an If VSL object, which tests surface:recursion channel value and activates sampling only at suitably low recursion steps. GI tracing depth can also be randomized by using a value computed by a Random VSL object as the input for the If block.

The post processing configuration for global illumination includes a component GI_brightness_control. As the name suggests, you can control the amount of indirect illumination using it.

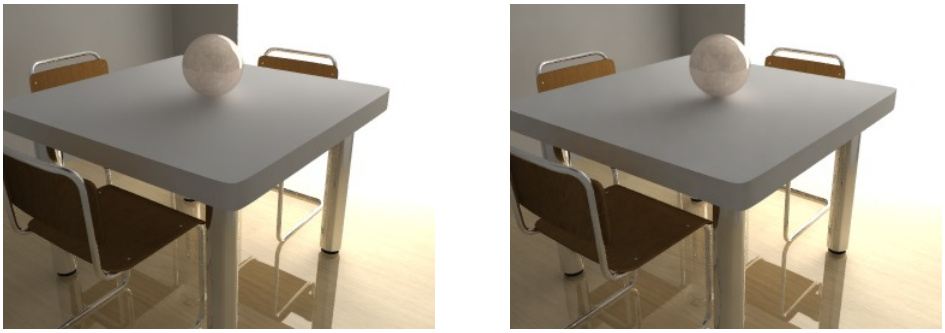


- *Reflected GI* - Adjusts the amount of global illumination visible through reflections and transparent objects. Usually the value should be slighty larger (10% - 20 %) than the end value of the control curve for diffuse GI, because the finite recursion depth of ray tracing darkens reflected GI slightly more than diffuse GI. A higher value compensates the difference.
- *GI Contrast* - The contrast option scales GI by the viewing angle from camera. Under 'flat' lighting conditions, i.e. when a surface gets a similar amount of illumination from all directions, shadows become weak or disappear. GI Contrast adds a compensating shading factor, which helps to preserve fine surface details such as bump maps.
- *Control curve of Diffuse GI* - With this gamma correction curve, you can, for example, enhance illumination at dark corners but leave the bright areas of the scene unchanged. To brighten indirect illumination everywhere, move the middle points and the points at the right end upwards. The start point of the curve should usually stay at zero level.

The VSL effect called **blurwidth_control** provides some additional control over GI filtering. When GI ray count is low (e.g. less than 10 rays per pixel) or idirect illumination is unusally noisy, there is not much change to regulate blurring, and the control curve should be close to full level 1.0 everywhere. However, if a large number of GI rays is used (several tens of samples per pixel), the blur width control curve can be adjusted towards a diagonally ascending line, as shown in the example image below. Such a control curve reduces filtering in strongly shadowed areas. This improves the contrast of GI remarkably.



Blur width curves for low sampled GI (left) and for a high number of GI samples (right).



The effect of the blur width control curves to global illumination. The image on the right side was filtered using a diagonal blur width control line as explained above.

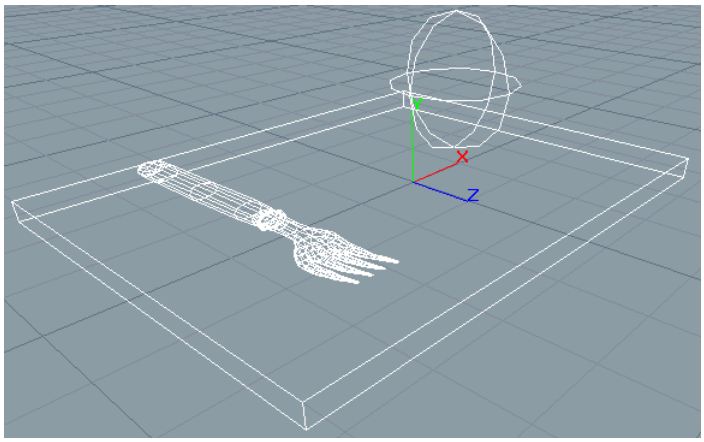
Outline Rendering

The next tutorial shows how to use outline rendering techniques of Realsoft 3D.

Tutorial level: Advanced

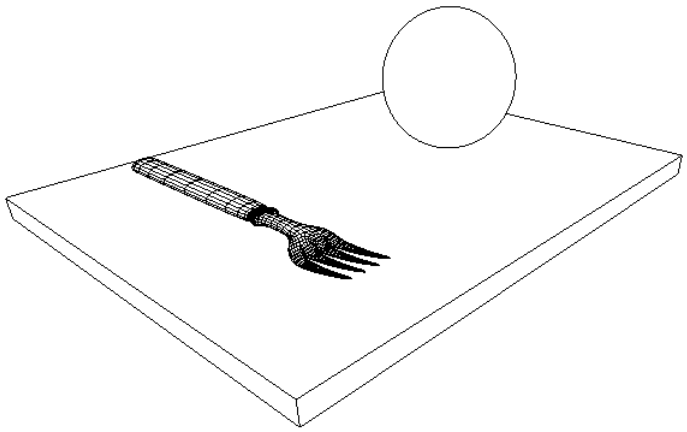
Example project:
tutorprojects/rendering/outlines

1. First we model a simple test scene: a flat analytic cube (table), a sphere and a freeform object modeled using the SDS tools (a fork). You may also load the test scene: 'tutorprojects/rendering/outlineobjects'.



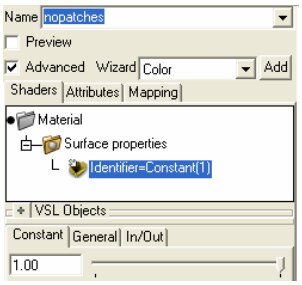
A test scene for outline rendering

2. Go to the Render settings tab of the Select window and drag&drop 'Black&white outline rendering' into the view window. Render the view - the result is a two-colored image.



Basic outline rendering

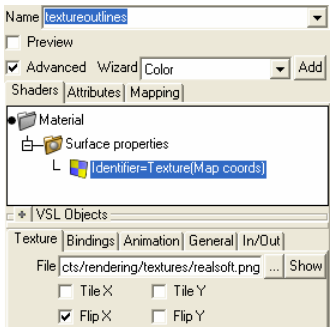
3. Let's remove the dense network of mesh boundaries from the fork object. Go to the materials tab of the select window and create a new, empty VSL material. Open the property window and rename the material as 'nopatches'. Check the Advanced option to see the VSL hierarchy. Also you may need to click on the '+' sign to access the VSL Objects. Drag & drop a Shader object into the root level of the material. Then drag & drop a Constant object into the Surface propertie shader. Set Output channel to Surface:Identifier using the popup menu of the VSL hierarchy tree. Change the assigned constant value to any nonzero value, for example 1.0. Identifier value 0 would blend the object with the background (empty space has identifier zero). If you later need to process several objects this way, just create several materials and assign value 1 in the first one, value 2 in the second one etc.



A material that hides surface patches in outline rendering

4. Make sure that the fork is selected and drag&drop the Nopatches material to the view window.

5. While we are in the material editor, let's create another material which texture maps some outlines. So, create a new VSL material called 'textureoutlines'. Check the Advanced option. Then apply the Texture map wizard. Select the Texture object from the VSL hierarchy and change the Output channel again to Surface:Identifier. Then pick a suitable texture file, such as 'textures/logo_bw'. Set the Flip X option. Close the property window. Then select the sphere from the view window and drag&drop the new material into the view.



A material that texture maps outlines

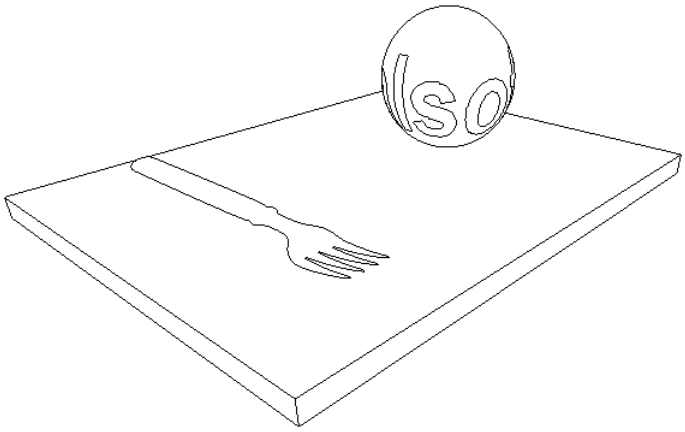
Black texture areas will again blend with the background, and texture areas where the first subchannel (=red) value is 1.0 will blend with the fork material (its identifier was also one). It is easy to fix such conflicts without changing the actual texture map; just add a unique integer value constant to the identifier channel after reading the texture value to it:

```
identifier = texture(map coords)
identifier += constant(999)
```

Now render the view window:

At this point, rendered outlines appear jaggy without antialiasing. We can improve quality by using a suitable post processing configuration.

6. Switch to the Post Image Effects tab of the Select window. Select New/Line from the popup menu to create a line drawing effect.



Customized outlines

7. Select Default effects from the effects list and open the property window. Drag&drop the new Line effect from the Available Effects list to the end of Active effects hierarchy. Then go to the Scaling tab of the property window. Set Output Scale X= 0.5, Output Scale Y= 0.5. In other words, image resolution will be halved after post processing. This also has the nice side effect of making the image antialiased. Close the property window.

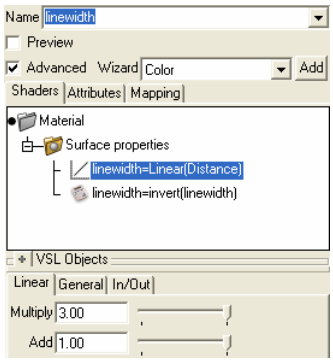
8. Drag&drop Default effects from the select window to the view, to make sure that it will be used in rendering. Switch to the Render settings tab of the select window. Open the property window of Black&white outline rendering. Go to the Post proc tab of the property window and make sure that the Safety area option is checked. Without safety area, Line effect will create gaps between render boxes. Then render the view. Rendering takes now longer but the quality is better. Outlines are now probably thicker, but you can adjust the line drawing width using the Min. Width attribute of the Line effect (Post Image Effects tab, property window of the Line effect).

9. As a final step, let's define varying line width. We need a channel and a material for that purpose. First go to the Channels tab of the Select window and add a new float channel called 'linewidth'.

10. Then go to the Materials tab and create a new VSL material. The contents of the material is shown in the example image.

The material simply defines that line width is dependent on the distance: i.e. width increases towards the camera.

11. Go to the object hierarchy tab and select the root level. Then switch back to the material library and drag&drop the linewidth material to the view. Since the material was applied to the root, all objects will obey the customized line width definition.

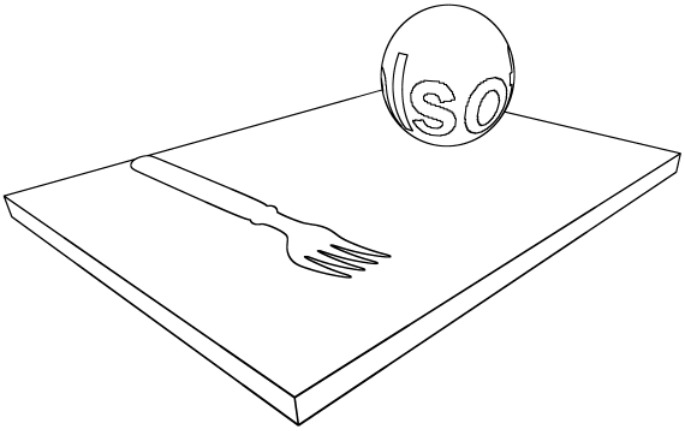


A material that defines line width

12. Switch to the Post Image Effects tab and select the line effect. On the property window, set Width Channel to the custom channel linewidth.

13. Select the Default Effects post image and decrease the output scaling factors to 0.25. This will improve antialiasing further so that the quality is good enough for final rendering. Drag&drop the Default Effects in the view window.

14. Output scaling to one quarter means that the program ray traces 16 times more pixels. To compensate the increased rendering work, select the render settings object Black&white outline rendering. On the property window, go to the Ray Tracing tab. Increase both Undersampling values to 8 pixels. Also, make sure that Interpolate is not set, it only slows the rendering without any result on the final output. Drag&Drop the render settings object Black&white outline rendering in the view window. Then render the view. If necessary, adjust Min. Width and Max. Width values of the Line effect which define the thickness range.



Antialiased outlines

Caustics

The caustics tools of Realsoft 3D compute how light reflects from shiny and transparent objects such as glass or shiny metal around the scene. Caustics can be computed into illumination maps or rendered as a part of the ray tracing process.

Adding caustics to your scene is usually easy. The first caustics tutorial about magnifying glass shows the basic steps.

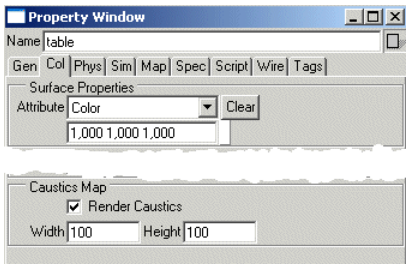
Tutorial level: Medium

Example project: tutorprojects/rendering/mag/glass

1. Model first the required test objects:

- An analytic rectangle is used as a table surface capturing the caustics
- An analytical cylinder forms the handle of the magnifying glass
- A Boolean and not operation between two analytical cylinders creates the rim around the lens
- The lens is an analytical sphere which is stretched suitably in one dimension
- A point light source is placed above the magnifying glass so that the glass will cast a shadow onto the table

2. Then map some materials to the objects: the standard glass material to the lens and a metal (steel, chrome etc.) to other parts of the magnifying glass. In the example scene, we have added a bump map to achieve a grooved handle. The bump map is basically two dense sine wave spirals, one traveling clockwise and the other counterclockwise around the handle. The bump map is not necessary for the caustics test, but it should be noted that the grooved shiny surface will create an especially interesting light scattering in the caustics lighting.



Object properties for caustics

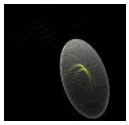
3. Select the ground surface (table) and open the property window. Go to the Col tab. Activate the Render Caustics option and define a suitable caustics map resolution for the table surface - 100*100 pixels is quite OK for our simple scene.

4. Before rendering, let's check the render settings. So, switch to the render settings tab of the select window and select a render settings object (for example reasonable quality). In the property window, go to the Ray Tracing tab and make sure that recursion depth is at least 3, preferably 4 or 5. The higher the level, the more accurate and detailed caustics you will get. Recursion threshold should be also quite low, for example 5.

Then switch to the Miscs tab and make sure that the Caustics and Automatic brightness scaling options are set. Default maximal brightness level is 1.0 which corresponds the usual white color - it is quite appropriate. The default sampling level 10 is low but enough to get us started. Close the property window and drag&drop the edited render settings to the view window to make sure that the view will use it.



The caustics options of render settings



*A 100*100 pixel caustics map*

5. Start rendering. As the progress indicator messages show, the program first renders the caustics maps. Then the usual ray tracing starts. The caustics maps are placed into the 'temp' folder of Realsoft 3D home directory. If your machine is not a 3 gigahertz Pentium4 system, you may have time to switch to the image tab of the select window, create an image object and use the property window to browse and view the image file named as 'temp/causticsXXX'. The program cleans up the caustics maps automatically after rendering.

That's it! You can continue experimenting with the scene - for example, double the table's caustics resolution to 200*200 pixels and increase render settings Sampling value correspondingly. The result is that you will see crisper caustics patterns and finer details. A good rule of thumb is that usually 10-20 sampling units in the Miscs tab of the render settings is required for every 100 pixels of caustics maps: 300 pixel caustics resolution - sampling=30, 500 pixels - sampling=50 etc. If there are several caustics targets in the scene, the maximal resolution defines what is the suitable sampling rate. Too low sampling will result to noisy caustics.



Magnifying glass is not totally focused on the table



Note

Caustics illumination is attached as a texture map to target objects. A rectangle is a very efficient target because it has only one flat surface and map pixels are not wasted for the backside. Note also that a freeform object (NURBS, SDS etc) must have proper UV coordinates initialized to function properly as a caustics target.

Caustics Mapping

The previous tutorial explained how to use the automatic caustics rendering feature of Realsoft 3D. Now we examine how to add caustics manually to an object as an illumination map. The process is the same as when mapping indirect illumination: the UVImage tool computes the caustics map, creates required shaders and attaches them to target objects using material mapping.

This method has the advantage that the caustics illumination map (and other parts of the scene) can be edited after caustics rendering. This gives additional possibilities to control the rendering process.

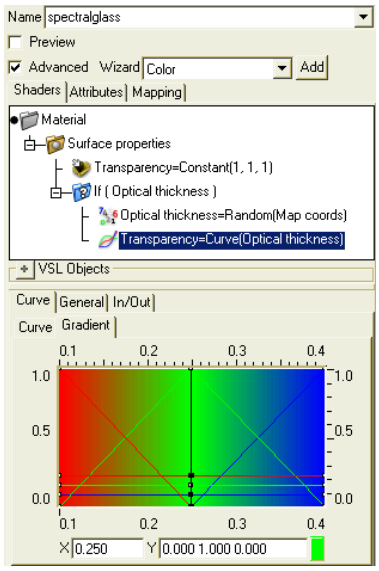
Tutorial level: Advanced

Example project: tutorprojects/rendering/spectralcaustics

1. Create a test scene: a flat analytic rectangle (floor) and a prism above the floor. Then add a point light source (a distant or spot light will also do the job) above the scene.

2. This time we will use a special kind of glass material: it processes multiple wavelengths in order to show the classical spectrum of white light which has travelled through the prism. Go to the material library, create a new VSL material called 'spectralglass' and open the property window. Define a material shown in the image below:

Let's examine the structure of the material in detail:



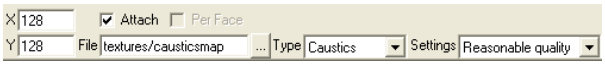
A glass material that traces multiple wavelengths

- The material is used only for refraction tracing. Therefore, the structure is maximally simplified to optimize rendering speed, and the root level contains only one Surface properties shader.
- In the surface properties shader, we first assign a full transparency 1,1,1.
- Next an If object examines if Exterior:Optical thickness is nonzero - in other words, if the examined ray is entering the glass material from an empty space. The wave length randomization is executed only when entering the glass, not in internal glass reflections. This both optimizes rendering speed and reduces sampling rate requirements, because each ray will use the same wave length during its traversal.
- If block starts with a random object that selects a random optical thickness value from a range that represents all sampled wavelengths. In our example, the range is quite wide, from 0.1 to 0.5.
- Each thickness value corresponds a wavelength which in turn can be seen as a certain visible color. Thickness values can be conveniently mapped to colors using a curve object.

3. Map the spectralglass material to the prism using a parallel mapping (the Random object needs a proper Map coords parameter).

4. Go to the Render settings tab of the Select window and activate for example Reasonable quality. Open the property window. Go to the 'Ray Trace' tab and make sure that recursion depth is at least 3, preferably 4 or 5. Recursion Threshold should be quite low, for example 5. Then switch to the Misc tab. Activate Caustics option if not yet activated and turn on Automatic brightness scaling. Increase the default sampling level to 20 - tracing multiple wavelengths requires lots of samples even for this simple scene. You can then close the property window.

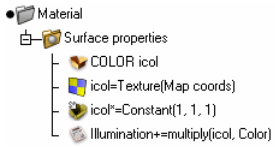
5. Select the floor rectangle from the view window. Then switch to the materials tooltab and activate the UVImage tool.



UVImage settings for caustics computation

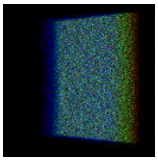
Set Type to Caustics, pick the proper render settings object from the Settings selector and enter a suitable name to the File gadget (for example 'textures/floorcaustics'). The default resolution of 128*128 pixels is quite OK. Hit Accept and wait until the tool finishes its job.

The tool adds a new material mapping object to the hierarchy and a new material to the material library. Check out the structure of the material using the property window. The structure is quite simple. The color from the illumination map is read into a variable, then multiplied with a constant value to scale the brightness and finally added to the total illumination.



A material that maps caustics to the floor

You can now select the texture object and hit the Show button to view the caustics map - it looks as in the adjacent example image. Quite promising! Before doing the first test rendering, let's modify the scene a bit.



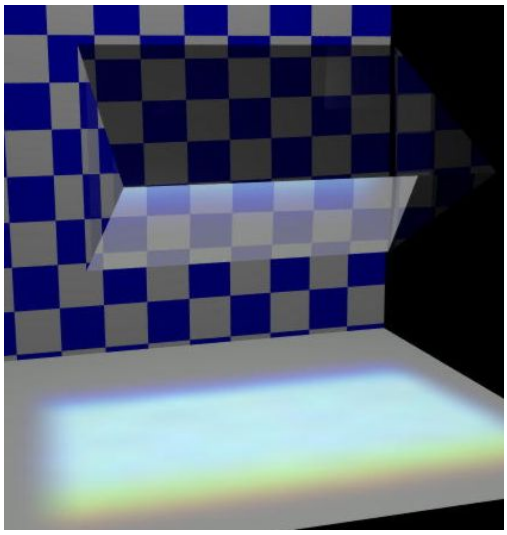
The caustics map created by a prism

6. The glass material we used was quite a special one, designed for caustics purposes. It would look quite bad in normal rendering. Therefore, go to the material tab and create a normal glass material if one is not yet included in the library. Then switch to the geometry tab of the select window, select the glass mapping from the hierarchy and change its target to the usual glass using the selector in the tool control bar.

7. You may also optimize the scene a bit now. Since the floor caustics already includes the shadow of the prism, you may apply one of the three optimization alternatives:

- Make the prism Shadow invisible on the property window
- The simple test scene has no other shadow casting objects as the prism and therefore you can also turn off Shadows completely from the render settings
- Add a Surface filtering shader to the normal glass material and assign transparency=constant(1,1,1)there

8. Render the view. If the spectrum looks noisy, go to the property window, select the Texture object of the caustics material and double its Blur value. You can also adjust the brightness at the same time if you like. The caustics map is precomputed and therefore new test renderings proceed quickly.



Light spectrum created by a prism

Animation with Global Illumination

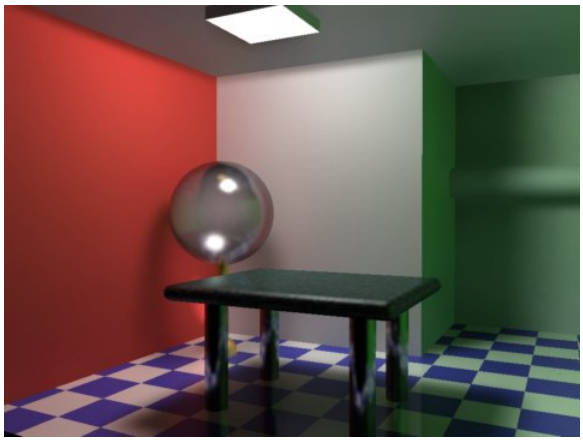
Computing the global illumination by random sampling light ray bounces is somewhat problematic for animations. Even when using a very high amount of light samples per pixel and heavy filtering, the computed illumination includes some random noise. Human eye easily notices even a low amount of noise. Just a couple of percents of noise from the full color intensity is still clearly visible. The constantly changing noise pattern usually appears quite unpleasant and distracting.

In this tutorial, we will examine how the noise problem can be solved using multi pass rendering. The big idea is to extend the illumination filtering over multiple frames. Also other random sampled shading components will be filtered in order to improve the quality.

To skip the details about how to configure multi pass rendering, go to the last section Multi Pass Rendering Summary. The summary tells you how to use the predefined components, which come with the installation.

Tutorial level: Advanced

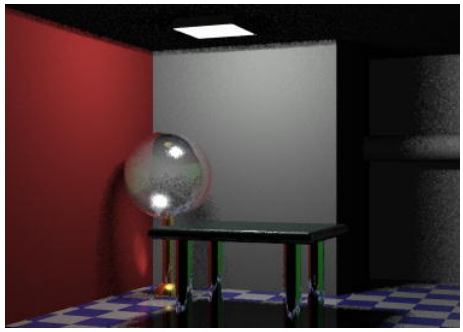
Example projects: tutorprojects/rendering/ gi_room.r3d, gi_post1.r3d, gi_post2.r3d



An example shot from the final animation

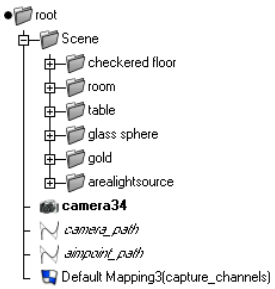
The First Pass: Ray Tracing Deep Files

1. Model a simple test scene or load the provided test scene 'tutorprojects/rendering/gi_room.r3d'. The test scene includes one area light source in the ceiling. Unless a high sampling rate and slow rendering is used, the shadows appear badly dithered. Nevertheless, you can set the area light's sampling rate to a low value, so that rendering time will be appropriate for animation use. If you model the test scene yourself, animate the scene for example by moving the camera along a path.



The test scene. The area light source is sampled with the minimal quality. Shadows are badly dithered.

2. We will apply post shading to the scene. Select the root level of the scene hierarchy. Go to the materials tab of the select window. select New/From Template from the popup menu and load in the file 'materials/illumination/capture_channels.r3d'. Then drag and drop the material to the view window to apply the material to the whole scene.

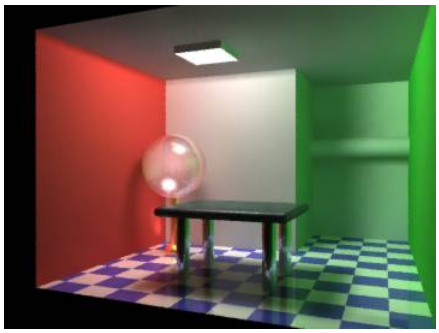


The hierarchy of the scene

3. At this point, switch to the post effects tab of the select window. First select Delete All from the popup menu to remove the effects we will not need. Then select the popup menu's Paste from a file and pick a file realsoft3d/posteffects/post_shading_effects.r3d. This loads two effect configurations, post_shading and post_shading_draft. These are the example systems for filtering shadows, reflections etc. - see the tutorial Post Processing/Post Shading for details. Drag and drop post_shading_draft from the select window to the view window.

4. Switch to the Render Settings tab of the select window. Drag and drop GI rendering to the view window. Make sure that View popup/Render/Backup Ray Trace option is enabled. Then select View/Render/Ray Trace. Adjust GI blurring and brightness parameters if necessary and apply View/Render/Post Process menu until the GI parameters seem to be suitable. You can also adjust other post shading components and shadow blurring until the shadows appear non-dithered. This is just a preliminary test stage - you can change all post processing parameters later. That is one of the main points of this tutorial.

So far, the workflow has been pretty similar to any project, which uses global illumination (or post effects in general). Next we prepare the first render pass. Instead of applying the post effects directly, we will collect all untouched raw data into very deep image files. This gives us a possibility to adjust the post effects later to produce variations from the same raw data. If the final animation shows some insufficient filtering quality, or if the customer suddenly wants a warmer lighting into the scene, this approach saves the most time consuming ray tracing phase.



Post shading applied to filter shadows and reflections

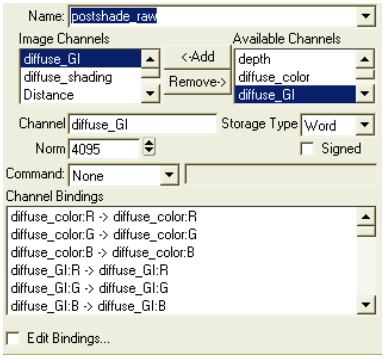
5. To define the deep image file contents, go to the output objects tab of the select window. Select New/Realsoft 3D from the popup menu. Make sure that the new format object is selected and open the property window. Use the name field to rename the format object for example as 'postshade_raw'.

We will have to add a large set of channels into this file format object. Select Distance from the Available Channels list and click Add. A new item labeled as Distance appears to the Image Channels box. Click the new image channel item and then change its Storage Type from Float to Integer. Very deep files consume huge amount of hard disk space. By optimizing the storage data type, we can reduce the file size. Word is usually

enough to store distances in a scene which has a diameter of some meters, Integer is sufficient for several kilometers and Float is enough for astronomical scales. After adding the distance, add other required channels - the full list is given below. Order of the channels does not matter.

- Distance - Integer
- distance2 - Integer. A second distance channel for reflected GI filtering.
- normal1 - Byte. The surface normal vector has unit length, therefore Byte is enough.
- normal2 - Byte. A second normal vector for reflected GI filtering.
- diffuse_color - Byte. The surface color value usually fits into a Byte.
- diffuse_shading - Word. Diffuse Illumination from light sources easily exceeds the Byte range.
- diffuse_GI - Word. Indirect diffuse illumination often has higher level than 1.0, therefore use Word.
- reflected_GI - Word. GI from reflections often needs higher bandwidth. Word is a good choice.
- postreflection - Byte. Reflected illumination. Byte is enough for this scene.
- posttransparency - Byte. Colors seen trough glass. We can use Byte here.
- self_illumination - Byte. Self illumination in this scene fits into Byte range.
- specular_shading - Byte. Specular highlights can exceed Byte range, but truncating to level 1.0 (white) does not harm in this project, because we will not utilize high intensity values for special effects.
- rn - Byte. Byte is always enough for the flashlight coefficient.

In the list above, the data types are mainly set to the minimal level in order to reduce the file size. If you have lots of free disk space available and feel uncertain about the right choice, just use one step better alternatives. Just remember that Float takes 8 times more space than Byte. The configuration below already consumes about 50 bytes per pixel. In the double PAL resolution, this means over 60 megabytes per image! A CD disk can only store about 8 of such images. For a 250 frame animation, you will need 20 GB of empty hard disk space. However, some of the channels are almost empty and contain only some local compact peaks, such as specular highlights. Therefore, the deep image files usually compress very well and a backup of the 20 BG raw data can most likely be stored to a DVD disk by using .zip compression.



Configuring the deep file format

The illumination components diffuse_GI, reflected_GI and diffuse shading are stored using Word data type. It may be possible to store them using Byte, if the lighting of the scene does not contain high intensity peaks. Insufficient data type increases sampling noise and therefore decreases rendering quality. If you use high intensity illumination effects, for example HDRI, bandwidth of the Byte data type is probably not enough.



Note

You can avoid the laborious file configuration step by saving predefined file format objects into a suitable folder. Realsoft 3D/imageformats/postshade_raw.r3d contains the required .r3i format object for this tutorial. Use Paste from a File popup menu to load it into the output objects library. You can check and optimize the data type for each channel if you like.



Note

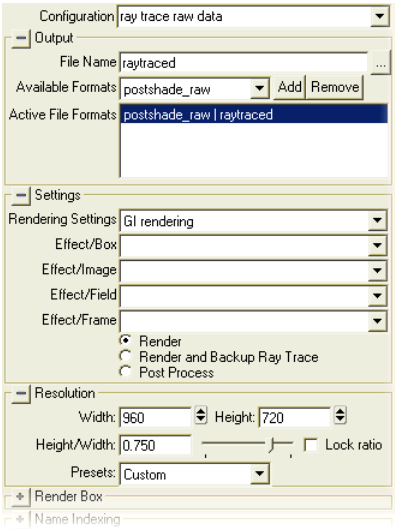
The file format defined above does not include all post shading channels. Illumination and no_fog are not stored, because the example scene does not include fog. The example version

postshade_raw, which can be found from the imageformats folder, is more extensive and includes all post shading channels. You can delete the mentioned two channels, as well as Glow size and Glow color channels to save some disk space. If you need these additional channels for other scenes, the recommended data types are:

- Illumination - Word. The total result of shading. If you need to spare disk space and don't need to trace illumination peaks accurately, use Byte.
- no_fog - Word. Illumination before a possible global fog. Use the same data type as for the Illumination channel.
- Glow color and Glow size - Byte.

When rendering your own projects, you may have to expand the channel collection even more, depending on the applied post effects.

6. When the deep file format is ready, open the file rendering window (Ctrl+r). Add the deep file format, which we configured above, to the Active File Formats list and remove other formats. Give a suitable name and path for the image files (for example 'raytraced' because these files will hold purely raytraced information without any post effects). Select GI rendering as the applied Render Settings object. Clear all post effect selections. Finally, set the resolution to the double of the intended final resolution. If you plan to create a 480*360 pixel animation, double it to 960*720 pixels. The extra resolution will be used for antialiasing purposes. Save the project as pass1.r3d and then start rendering.



Rendering setup for the first pass

The Second Pass: Post Shading and GI Blurring Single Frames

With the second rendering pass, we will filter the noisy area lighting and the global illumination. GI, which is the noisiest and most problematic shading component, will not be added to the shading yet. It will be stored for another, multi-frame filtering pass.

7. Go to the geometric objects tab of the select window. We do not need the geometry in the second rendering pass, so you can either delete all objects or make them invisible in ray trace rendering. However, it is a good idea to leave the camera object intact, because some post effects (DOF, lens flares) depend on the camera. Although we do not use such effects in this tutorial, this is a safe working habit to learn.

8. We can optimize the render settings a bit. Go to the render settings tab, duplicate the GI rendering object and rename the copy as 'QuickRT' or something similar. Using the property window's Ray Tracing tab, set Horizontal and Vertical Undersampling to 4*4 pixels and turn Interpolation off. Switch to the Distr tab and turn box rendering off. In the Post Proc tab, turn the Safety area option off (an additional continuity area around the image would not improve anything, because the rendering data comes from the deep files, which do not cover the extension region). This kind of render setting configuration initializes the empty black image in a minimal time. You can also load in the optimized settings from the 'realsoft3d/rendersettings/QuickRT' file.



Note

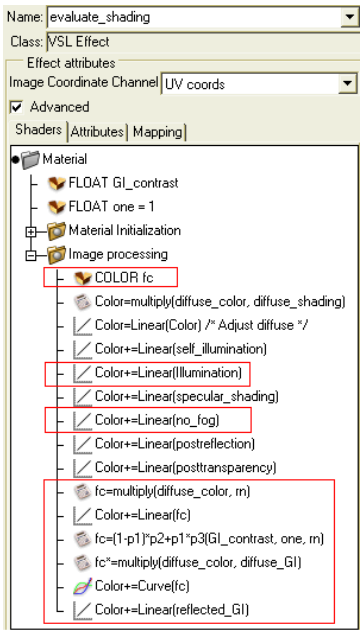
The render settings object above uses only one single rendering thread. Because of filtering continuity issues, use of several threads would not speed up rendering much. Later, when using frame blur effects, hard disk IO for several rendering threads reading simultaneously the same, very large data files might actually slow down the rendering.

9. Go to the Image Effects tab. We will now configure a post effect system, which filters the shading information from the deep ray traced files.

10. Because the filtering system will include the post shading components, one of the existing objects is a good starting point for the new system. So, select post_shading_draft and rename it as 'PostShadeFile'.

11. You can optimize the post effect system slightly to gain some rendering speed:

- Select 'remove_fog_and_custom_shading' from the Active Effects list and Apply Remove from the popup menu. There is no fog in this scene.
- Select the 'blur_fogginess' from the Active Effects list and remove it, too.
- You can remove 'Render glow material properties' and 'Camera defined depth of field' as well. There is no glow in the scene and DOF is not used (when applied, DOF should be the last effect of the final rendering pass).
- On the select window, select the VSL effect 'evaluate_shading'. On the property window, put the effect to Advanced mode and delete the following VSL objects from the shader tree:
  - Color fc - unused variable
  - Color+=Linear(Illumination) - custom shading not used
  - Color+=Linear(no_fog) - fog shading not used
  - fc=multiply(diffuse_color, rn) - no flashligt needed
  - Color+=Linear(fc) - no flashligt needed
  - fc=(1-p1)*p2+p1*p3(GI_contrast, one, rn) - GI will not be added in this pass
  - fc*=multiply(diffuse_color, diffuse_GI) - GI will not be added yet
  - Color+=Curve(fc) - GI will not be added yet
  - Color+=Linear(reflected_GI) - reflected GI will not be added yet



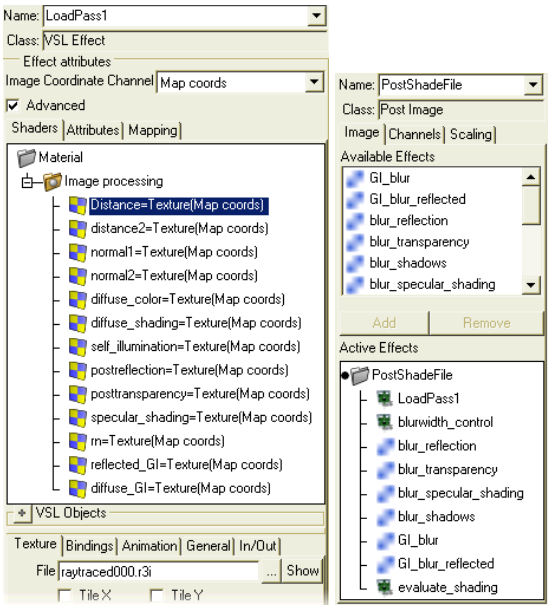
Remove inactive post shading components

12. One important effect is still missing: the one, which loads the channels to be processed from the deep files. Apply New\VSL Effect from the popup menu of the select window. Rename the new effect as 'LoadPass1'. Check the Advanced button of the property window. Activate the root node Image Processing and select New/Texture from the popup menu. Then use the browse button to select the first image of the deep image sequence. Switch to the Animation tab of the property window and change the Animation Type to Automatic. Open the popup menu of the VSL tree and change the Output parameter to first one of the required channels, Image:Distance.

13. Duplicate the first Texture object of the VSL tree using the popup menu. Change the output parameter to the next channel, Image:distance2. Repeat this for the remaining channels: normal1, normal2, diffuse_color, diffuse_shading, specular_shading, postreflection, posttransparency, self_illumination, diffuse_GI, reflected_GI and rn. Reading order does not matter.

14. Select again the effect PostShadeFile. Drag and drop the new VSL effect 'LoadPass1', which we just created, to the beginning of the Active effects list. The post effect setup for the second pass is now ready.

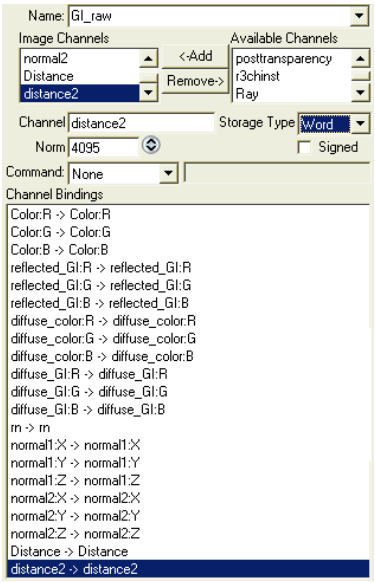
15. Go to the output objects tab of the select window. Select New/Realsoft 3D from the popup menu to add a new native format object (you can also load imageformats/GI_raw file to skip this step). Rename the new object as 'GI_Raw'. Then add the following channels to the object the same way as instructed in the step 5 above:



Define a VSL effect, which reads channels, and add it to the beginning of 'PostShadeFile' configuration

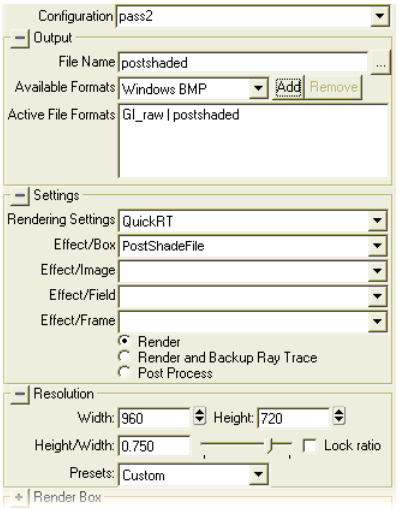
- Color - Byte
- Distance - Integer
- distance2 - Integer
- normal1 - Byte
- normal2 - Byte
- diffuse_color - Byte
- diffuse_GI - Word
- reflected_GI - Word
- rn - Byte

The new format no longer includes majority of the post shading channels, because they are already composited into the Color channel. The channels above store the GI information for the final filtering pass.



The file format which stores the channels for the final pass

16. Open the file rendering window. Change the file name and path so that images from the first pass won't get accidentally overwritten. Change the active file format to the new GI_raw object. Set the Render Settings selection to the QuickRT object, and Effect/Box to PostShadeFile. The resolution remains the same (=the final resolution doubled). Save the project as 'pass2' and start rendering the animation.



Rendering the second pass

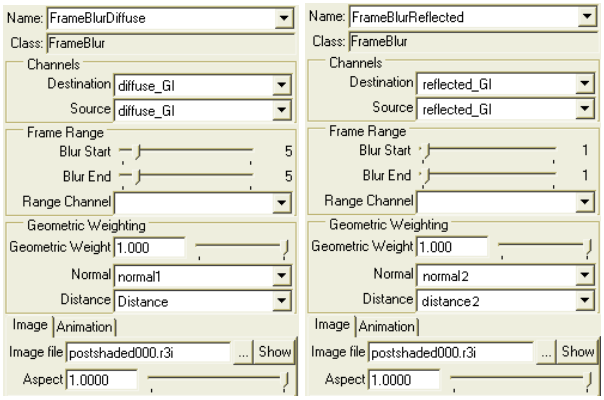
The Third Pass: Multi Frame Filtering

17. When the second pass rendering is finished, go to the post effects tab of the select window. We will now configure the last filtering system. An example system is included in the file 'realsoft3d/posteffects/MultiFrameGI.r3d' - load it in to skip the steps 18-21 of the tutorial.

18. Select New/FrameBlur from the popup menu. Rename the created effect as FrameBlurDiffuse, because we will blur diffuse global illumination with it. Use the browse gadget to select an image from the second render pass. Switch to the Animation tab of the property window and change Animation Type to Automatic. Go back to the Image tab and set Blur Start and Blur End to 5 frames. This means that the effect will combine GI information from 5 frames before the current frame to 5 frames after the current frame, 11 frames total. In each frame, we will get ten times more information about the global illumination; this will help to reduce the noise significantly.

Set Source and Destination Channels to diffuse_GI. This means that the effect will read diffuse channel from the disk files and store the blurred result to the diffuse channel of the post processed image. Then set Geometric Weighting to 1.0 - frame blur should combine GI information only from geometrically continuous areas.

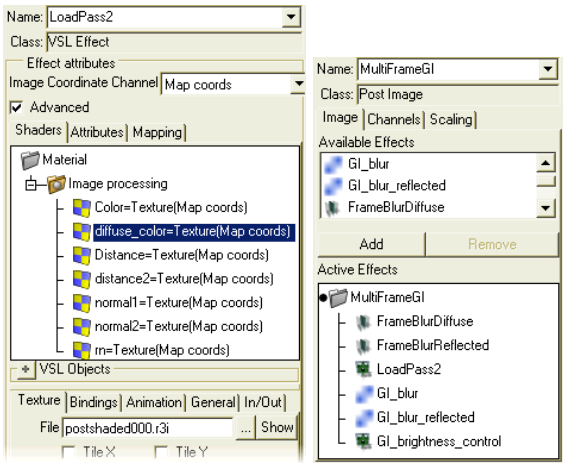
19. Duplicate the created FrameBlur effect. Rename the new effect as FrameBlurReflected. This time set the source and destination channels to reflected_GI. This part of the GI is usually much more coherent as the diffuse GI. Therefore, set Start and End to only 1 frames.



Frame blur effects for the diffuse and reflected GI

20. Create a new VSL effect and rename it as LoadPass2. Add 7 Texture objects, which read Color, diffuse_color, normal1, normal2, rn, Distance and distance2 from the pass 2 files. Remember to set the animation type of each Texture object to Automatic. Note that we do not have to read diffuse_GI or reflected_GI channels here - the two frame blur effects take care of that.

21. Duplicate the post effect PostShadeFile and rename the copy as 'MultiFrameGI'. Remove all effects above GI_blur from the MultiFrameGI configuration. Add the above created effects FrameBlurDiffuse, FrameBlurReflected and LoadPass2 to the beginning of it in this order. Leave GI_blur and GI_blur_reflected to the configuration. These effects will be used now again in order to eliminate illumination banding artifacts, which easily occur, when a moving camera reveals previously obscured areas of the scene (GI sampling within a frame is heavily randomized, whereas time stepping is not random at all but happend at regular intervals).



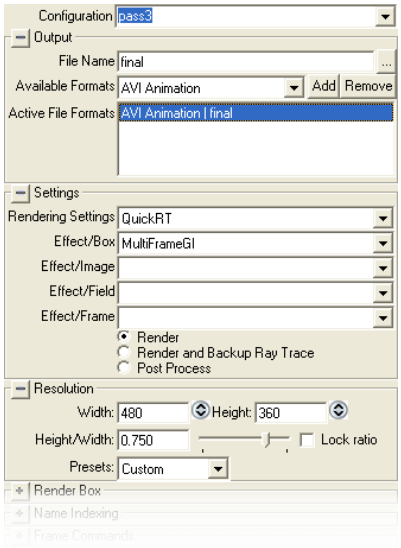
The VSL effect, which loads channels from the second pass images, and the post effect configuration for the final pass

So, we first blur GI over image area, then over time and finally again within each frame. However, you can and should reduce GI_blur values quite much for the second filtering round. For example, GI_blur Level can be 10 and Iterations 2. Similarly, GI_blur_reflected can be reduced to Level=5, Iterations=1.

Contine editing the contents of MultiFrameGI as follows: remove all effects after GI_blur_reflected. Then add GI_brightness_control to the end of the MultiFrameGI configuration. It will be the last effect. It adds the many times filtered global illumination to the images.

Select MultiFrameGI effect and go to the the scaling tab in the property window. Set Output Scale X and Y to 0.5. The last pass will shink resolution to one half, thus improving the anti-aliasing quality.

22. Open the file rendering window. Now set the Active Outputs to a suitable 'normal' image or animation format, such as .jpg or .avi. Define also a suitable file name. The render settings should be again QuickRT. Set Post effect/Box to the new MultiFrameGI. Drop the resolution to the actual final resolution (e.g. 480*360) from the doubled value. MultiFrameGI, which has Output scaling factors 0.5, halves the resolution from its source data to antialiase the images. Save the project as pass3 and start rendering.



The final rendering pass

Multi Pass Rendering Summary

A brief summary of the multi pass rendering steps using the predefined components is given below. You can naturally use your own components instead of the objects, which come with the standard installation.

1. Model the scene as usual

2. Default map the material 'realsoft3d/materials/illumination/capture_channels' to the root level.

3. Render using the deep file format 'realsoft3d/imageformats/postshade_raw' in doubled resolution and with no antialiasing.

4. Load in the post effect system 'realsoft3d/posteffects/PostShadeFile'. Select the included VSL effect 'LoadPass1' and using the property window, change the file name of every Texture object to the image sequence you rendered in the previous pass. Set the frame count in the Animation tab of each Texture VSL object to match the frame count of yor animation. Remove or disable the VSL object, which adds filtered GI to the image. Change Output Scale X and Y of the LoadPass1 configuration to 1.0.

5. Make all geometry objects except the camera RT invisible and render the second pass to 'realsoft3d/imageformats/GI_raw' format with an optimized render settings object 'realsoft3d/rendersettings/QuickRT' and 'PostShadeFile' as the post effect per render box.

6. Load in the post effect system 'realsoft3d/posteffects/MultiFrameGI'. Update the image file names of the two FrameBlur effects to match the image files from the second pass. Change the Frames value of the FrameBlur effects in the Animation tab. Then update the file names of the VSL Texture objects in the VSL effect 'LoadPass2', and set the Frames value in the Animation tab to a correct value.

7. Render the final pass to the desired format with the QuickRT settings and the MultiFrameGI post effects.



Chapter 9. Post Effects

Post Image Effects

The Realsoft 3D rendering engine includes an open and powerful post processing system. Effects such as glow, lens flares and dust can be added to the images using the included post effect modules. The powerful features of the post processing engine include:

- **Post processing is freely configurable**. The user can define any combination of effects in any order. The same effect can be used multiple times with different parameters within the same rendering pass.

- **Expandable data channel configuration**. The user can define and use an unlimited amount of data channels to achieve desired effects. All channels used for defining material and object properties are automatically available for use in post processing. For example, the user can define a custom glow casting light source. The glow effect can add glow to those parts of objects, which are illuminated by the special glow light (although this light may be otherwise invisible). All channels can be output to any suitable configuration of file formats: Color+Alpha ->Targa RGBA, Distance+Transparency+Reflection->.bmp, etc.

- **User definable data accuracy**. The post processing system support all data bandwidths from the traditional byte/channel to the full double precision float/channel accuracy, which is certainly enough for any purpose.

Most post effects are quick and memory efficient to render. Note that they cannot fully interact with ray tracing, because they are rendered after the raytracing pass.

Post processing is an optional step in the rendering pipeline. If post processing is not activated, the rendering engine renders the images directly to the actual output device, such as a file (BMP, TGA, AVI, etc.) or to a view window. When post processing is activated, the image data is copied to a *Post Image*. There can be any number of *Post Effects* associated with the post image; each will add its contribution to the result. When all the effects have been applied to the post image, its contents are copied to the actual output device(s).

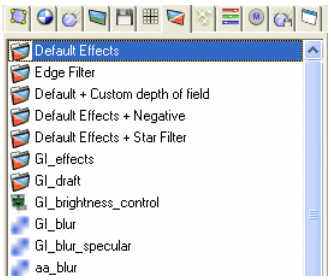
Managing post effects

Post effects can be divided into two groups: effects which are used for rendering particles ('Particle effects') and effects which are applied to the entire image ('Image effects').

Image effects can be created and managed through the select window's Post Image Effects tab.

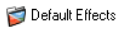
This tab includes two kinds of objects:

- Post Images
- Effects, which can be applied to post images



Select window's Post Image Effects tab

Post images provide the necessary image buffers for post processing.



A post Image

Post effects are the actual workhorses. Any number of them can be attached to a single post image using the property window.

The most common post effects are rendered automatically. Namely, the default user interface is configured so that the file rendering window and the view windows use a post processing configuration defined by the post image called Default Effects. This configuration renders all particle effects, glow and depth of field. You can select another configuration from the file rendering window or from the view property window.

This is the basic theory you need to know in order to use post processing in Realsoft 3D. Let's go through one example demonstrating how to add post effects to your images in practice.

Rendering post effects to a view window

We will experiment using a simple post effect called Edge Filter. It examines sudden color changes in the image to detect edges of objects, and changes a shaded image to an outline drawing.

Tutorial level: Beginner

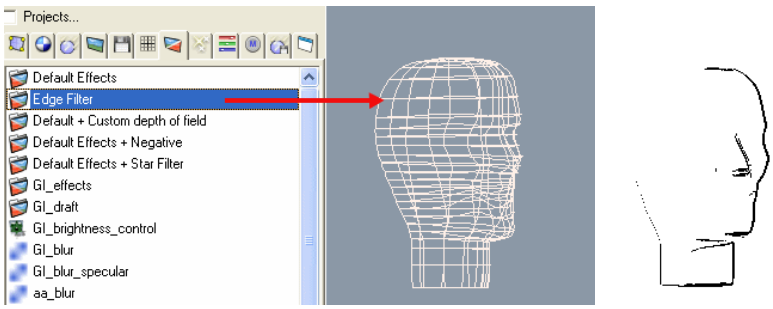
1. Model a scene, or load a provided example 'models/nurbs/simple head'. Render the view; you get the usual shaded result.

2. Select the Post Image Effects tab from the Select window. Drag and drop the Edge Filter post image to the view window.



Note

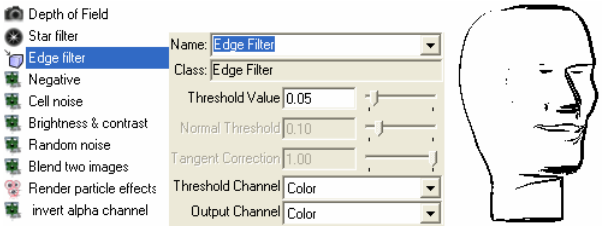
Drop the post image object. There is another 'Edge Filter' object appearing lower in the list, but it is a sub object of the post image. Remember that you always need one post image to set up post processing.



Selecting post effects for a view using drag & drop. The result rendered on the right.

3. Render the view. The edge filter changes the shaded image to an outline image.

4. The amount of detected edges depends on illumination, objects and effect settings. We can try to extract more edges. Double click the actual Edge filter effect (the lower one) on the select window to open the property window. Reduce the Threshold value to, say, 0.05, and render again.

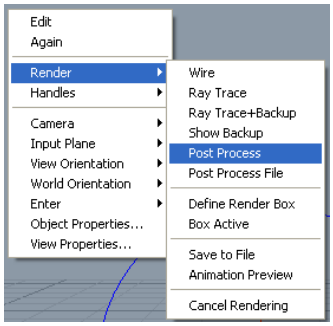


Adjusting the edge threshold value

Every time you re-render the scene, the rendering engine also ray traces the shaded image. If the scene is complex, this can make experimenting with post effects slow, because ray tracing usually takes much more time than the actual post processing. Fortunately there is a faster way:

5. We will continue experimenting with the edge filter: adjust a new Threshold value from the property window. Now, select Ray trace+Backup from view window's popup to render the image.

6. Now you have a new test image on the view and the rendering backed up. Adjust the threshold value once more. Then, instead of selecting full render, select Render/Post Process from the view popup. Only the post effects are rendered this time, and the result appears much quicker.



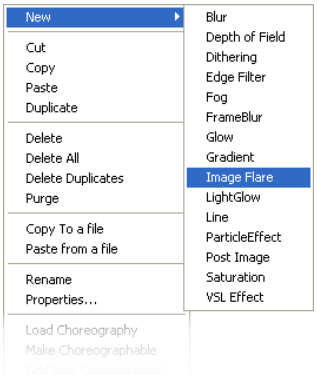
Rendering post effects only

Configuring post effects

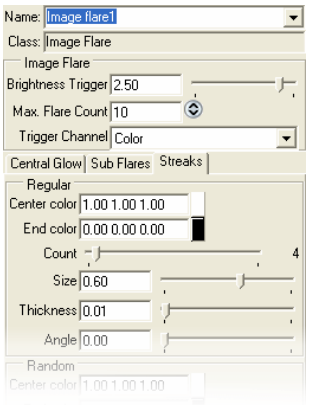
In this example, we will set up a post effect configuration, which applies a star filter effect. The brightest areas of images will get shiny star-like highlights. Such an effect is already available in the default configuration and you can activate it the same way as Edge Filter in the previous tutorial. Anyway, we will start from scratch to get familiar with the post processing tools.

Tutorial level: Beginner

1. Go to the Post Image Effects tab in the Select window and create an Image Flare object (right click and select Image Flare from the popup as shown in the image).

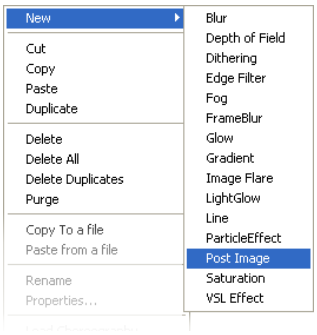


Create an Image Flare object



Specify desired image flare options

2. Double click the created Image Flare object in the select window. This opens the property window allowing you to define image flare specific options. Set the Brightness Trigger value to 2.5. The trigger value is the sum red+green+blue. For example, brightness of pure white 1+1+1 = 3 exceeds the trigger value. e.g. Any value that is within the trigger threshold receives an image flare. A value of 3 limits the image flare to only pure white pixels. 2.5 allows any reasonably bright pixel to fall within the threshold.

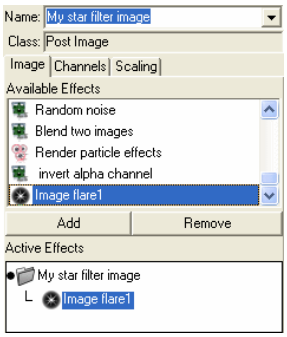


Create a new post image object

3. Leave the property window open and create a Post Image from the select window's popup menu.

4. Using the property window, rename the created post image to say 'My star filter image'.

5. The property window shows you two lists. The upper one includes you all effects available (including the Image Flare object you created above). The lower Active Effects list shows you the effects applied to the post image. Drag & drop the Image Flare object from the upper list into the lower list.



'My star filter image' contains only one effect, 'Image flare1'



Note

You can remove inserted effects by selecting Remove from the pop-up menu of the Active Effects list

6. Switch the select window to show the Rendering Settings tab. Select the settings, which the view window is using (or drag & drop 'Reasonable quality' to the view). Double click the Reasonable quality setting to bring up the properties window. Switch to the Distrib tab and disable the Box rendering option. Image flare has an exceptional property: it works better when box rendering is disabled. The reason is that Image flare analyzes continuous bright areas and adds the flare to the brightest position of each area. Render boxes may break these continuous bright areas.

7. Close the property window. Model a couple of objects and add a light source or two to the scene, so that the Brightness Trigger limit will be exceeded.

8. We have not yet assigned the new post processing configuration to any rendering engine. In the previous example, we did it by dropping the post image to a view window. This time we will use the view property window. Open it from the View Properties.. popup menu of the view window. Go to the Render tab and select 'My star image filter' from the Effect/Box gadget in the Rendering Options frame. Render to see the result.

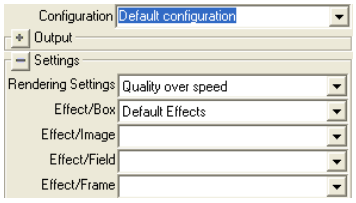


Image flares

Post processing configuration for file rendering

To activate a post processing configuration for file rendering, select a desired post image from the Effect/Box list. The post image will then be applied to each rendered image box. You can also define a post processing configuration that is applied to complete images , animation fields and animation frames.

If you don't want post effects to be rendered, clear these fields.



A post image 'Default effects' selected



Note

All selected effects are applied regardless of animation settings. For example, Effect/Field is processed even if field rendering is switched off.

Post Particle Effects

The previous post processing tutorials introduced some post image effects, which examine and modify the image rendered by the ray tracer.

The program also includes another category of post effects: so called post particle effects. They also draw their effect to a post image during post processing. The characteristic property of post particle effects is that they use geometric objects (particles) to define the position of effects in 3d space. The following comparison illuminates the difference between the two effect categories:

- 'Image flare' *post image* effect draws flares to the brightest spots of the image
- 'Lens flare' *post particle* effect draws flares to the positions defined by the particles

Mapping effects to particles

You can define the appearance of post particles using particle effects just like you define the appearance of ordinary geometric objects using materials: by mapping effects to particles. The idea is that particle objects define geometric properties such as position, size and orientation. The mapped effect uses the geometry information when rendering a shape representing the particle. The following example, a flock of lens flares, demonstrates this.

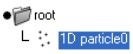
Tutorial level: Beginner

1. Create a bunch of 1D particles using the particle tool. Activate the tool from the Creation tab of the toolbar and drag the mouse around the view (left mouse button held down). Click the Accept button to finish the particle tool.



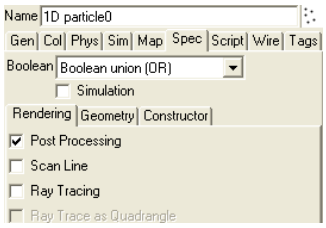
Particle tool activated

2. Render the view. You can see the particles as small dots. However, there is no post processing going on yet - what you see is *scanline* particle rendering.

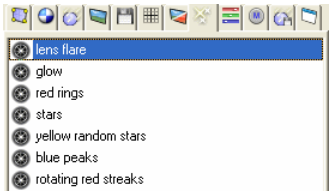


Particle object selected

3. Make sure that the particle object is selected. Open the property window from the Object Properties.. entry of the view popup menu, go to the Spec tab and clear the Scan Line option. Now the particles will be rendered using post processing system only. As a matter of fact, some post effects are sensitive to object distances, and leaving scanline drawing enabled may disturb post particle drawing later (post particles become hidden by scanline particles). Close the property window.



Scanline rendering disabled

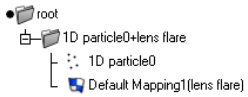


Particle Effects

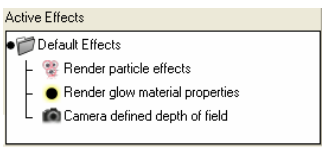
4. Go to the Post Particle Effects tab of the select window and drag&drop the desired particle effect into the view window. This maps the effect to the selected object (=particles).

5. You have now created a flock of 'lens flare' particles. Go to the object hierarchy tab in the select window, and open the sub hierarchy of the new object level '1D particle+default mapping(Lensflare)'. The hierarchy should look something like in the example image:

Note how object hierarchy is used exactly the same way as in material mapping (a wooden candlestick consists of the candlestick and wood mapping).



'Lens flare particles' object consists of two sub objects: 1D particles and a lens flare effect

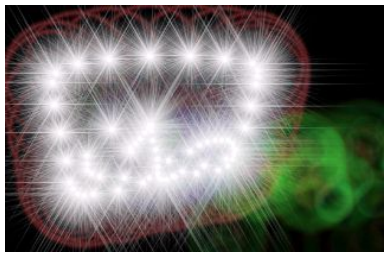


The effect configuration contains the module for rendering post particles

6. Everything should now be ready for rendering the flares. However, let us check one important setting before rendering. Go to the Post Image Effects tab of the select window. To make sure that we are using the right post processing configuration, drag & drop the Default Effects item to the view window. It is used by the standard startup project, but you may have changed the configuration when experimenting with previous post image effect tutorials. Then double click Default Effects to open the property window. The window shows the contents of the selected post image.

There is one particularly interesting item included in the list of active effects: Render particle effects. It must be included or otherwise post particles will not be rendered. By changing the position of this object in the list, you can define when the post particle effect rendering takes place. The configuration is suitable, so close the property window without changes.

7. Click the ray trace button in the view control bar (or select the view's popup menu Render/Ray trace).



A flock of lens flares

Using multiple effects

Let's assume you want to create two particle flocks: one whose particles are rendered with a 'star' effect and another one, which is rendered with a 'red rings' effect. To do this:

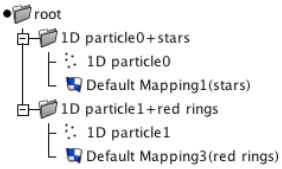


1. Activate the particle tool and create a number of 1D particles. When done, select Accept from the view's popup menu.

2. Activate the particle tool again and created the second particle object. The two particle objects might look as in the example image.

3. Select the first 'stars' particle object and drag and drop the 'star' effect from the select window's Post Particle Effects tab into the view window.

4. Select the second 'ring' particle object and drag & drop the 'red rings' effect into the view window. The current object hierarchy is shown on the right.



Two particle objects with different effects

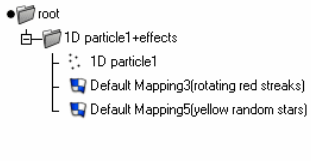
5. Start rendering.

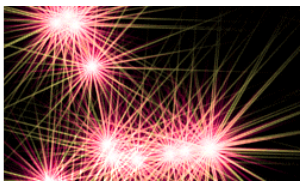


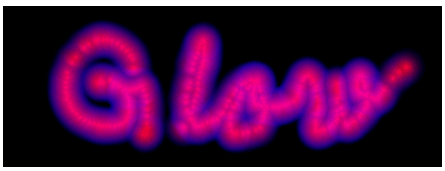


Note

You can combine effects by mapping multiple effects to a single object. The order of mapping objects in the hierarchy defines the order the effect are rendered.









Creating new particle effects

New particle effects can be created from the popup menu in the Select window's Post Particle Effects tab. The New sub-menu shows you the installed post effects. Just select an entry to create a new effect and use the property window to define its attributes as usual.



Note that one effect, such as the Lens Flare, can render a large number of different kinds of effects, from simple stars and rings to complex lens flares.

Blur effect

By default, Realsoft 3D's render engine computes razor sharp images. Sometimes it is desirable to produce 'smoother' images - to obtain a certain image style or perhaps to reduce antialiasing problems. Smoothing is a quick way to improve quality when only a low level of actual ray tracing antialiasing can be used (for speed reasons etc.).

The Blur effect is a useful tool in this situation. The fact that blur can be applied selectively greatly enhances the usability of the effect.

Tutorial level: Medium

Example project: 'tutorprojects/postprocessing/selectiveblur'

1. The first step is to model a simple test scene. For example, create a large floor plane using the rectangle tool and place some test objects above it. Texture map the floor and take a suitable view to the scene so that texture pattern gets dense in the horizon. Then ray trace the view.

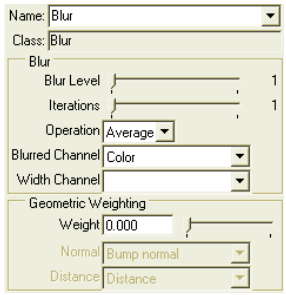


A simple test scene for the blur effect



Blur with default settings added to the image

2. Go to the Post Image Effects tab of the Select window. Select New/Blur from the popup menu. Then drag and drop the new effect that appeared to the end of the effects list to the view window. This inserts the effect to the current effect configuration. Render to see how the blur effect changes the image.



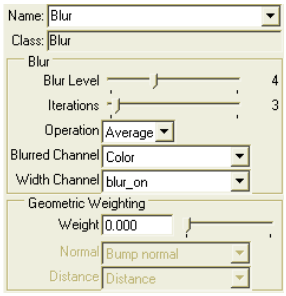
The default blur settings

3. Select the blur effect and open the property window. By default, the blur effect used the minimal amount of blur (level 1, iterations 1). The blur was applied uniformly to all parts of the image.

Let's experiment with selective blurring. For example, we may try applying the blur only to those parts of the image, which are covered by the floor pattern. Other (foreground) objects remain sharp.

4. Switch to the User Definded Custom Channels tab of the select window. Select New/Float from the popup menu. Rename the created channel object as 'blur_on'.

5. Go to the Geometric Objects tab of the select window and select the floor object. On the property window, select the second Col tab. Then select blur_on channel from the Surface Properties/Attribute list. Define the value 1.0 to the attribute value field.



Settings for selective blur

6. Now switch back to the Post Image Effects tab of the Select window. Make sure that the blur effect is selected, and select blur_on to the Width Channel gadget of the property window. You may also increase the Blur Level and Iterations values a bit.

7. Render the view again. Now the floor is strongly blurred, whereas the foreground objects remain fully sharp.



Selective blurring

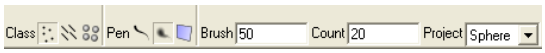
Particle Smoke

A large amount of post processed particles can be used to simulate smoke. The following tutorial shows how to do this.

Tutorial level: Medium

Example project: 'tutorprojects/postprocessing/smoke'

1. The first step is to model a cloud of particles. Activate the Particle tool from the Creation toolbar and select the airbrush options like shown below. Drag around the view window to draw an irregular cloud of particles. Select Accept to finish the tool.



Options for drawing a 1D particle cloud



The smoke scene

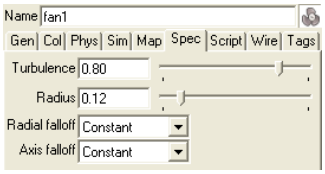
2. Go to the Simulation toolbar and activate the Fan tool. In the view window click below the particle cloud and click again slightly above to define a vertical axis for the fan (you might want to hold down the shift key to enable movement locking). Define an appropriate width for the fan.

3. To achieve a more interesting animation, place an object above the particle cloud. So go back to the Creation toolbar and activate Analytic Sphere from the Analytic tab. Create a sphere above the particles.

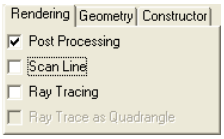
4. You may also add a light source to illuminate the scene (use Light Sources/Point from the Light Sources toolbar).

The scene should now be similar to the adjacent illustration.

5. Next you have to adjust some object properties. Go to the select window and activate the fan object. Open the property window from the popup menu. Go to the Spec tab and set Turbulence to 0.8. The other default settings are OK for our example.



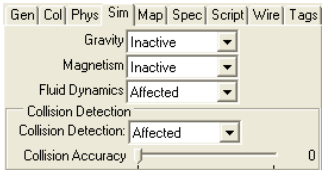
Defining fan turbulence to 0.8



Turning off Scan Line

6. Select the 1D Particle object from the select window. The property window shows three sub tabs in particle's Spec options. Clear Scan Line from Rendering options. You may also check the Geometry tab. It shows the amount of particles in the Count field. You will need several thousand particles. You can easily increase the amount by typing the desired value to the Count field.

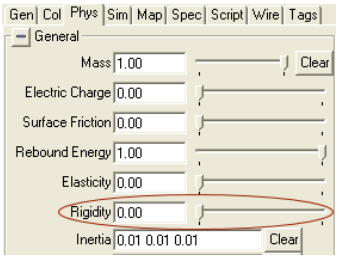
7. Switch to the Sim tab of the property window (the particles should still be selected). Set Collision Detection to Affected, because we want the particles to travel around the sphere, not through it. Set Fluid Dynamics to Affected, so that the airflow from the fan will move the smoke.



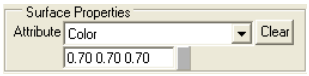
The simulation options of particles

8. Switch to the Phys tab. Change the Rigidity slider from 1 to zero. This allows individual particles to move freely instead of being a rigid object.

9. Switch to the Col tab. The Surface Properties frame includes a property list and a value gadget for the selected property. First select a suitable color for the smoke, for example a light gray. Then select Fade form the Attribute list and define a high fade value between 0.90 - 0.99. The more particles you created, the more transparent one single particle can be. Do not set fade to 1.0, because then you would not see any particles at all.



Zero rigidity allows particles to move freely in airflow





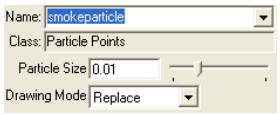
Defining the color and fade of particles

10. The particles are now ready. Next select the sphere. Go to the Phys tab of the property window and increase the Mass value to 1000 kilograms. The smoke should not move the sphere which would happen if the sphere is very lightweight. Alternatively, we could also make the smoke very light.

11. Go to the Sim tab and set Collision Detection to Cause to enable collisions with the particles.

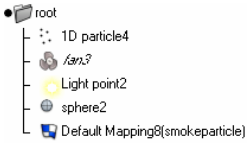
12. Now select the root level of the object hierarchy. Activate Simulation from the tool control bar  icon or from the Spec tab of the property window. This instructs the program to compute physical simulation according to the properties we defined above for the selected hierarchy level.

13. Select Post Particle Effects from the select window. Select New/Particle Discs from the popup menu. The property window now shows the default properties of the effect. Change the name of the effect to 'smokeparticle'. The other properties are quite suitable: each particle has a diameter of 2 millimeters, which should work well. The drawing mode is Replace, which is also OK, because smoke hides objects (i.e. the color of smoke replaces colors of the objects behind it). When creating fire, a more suitable drawing mode would be Add. Now close the property window.



The post effect for drawing smoke particles

14. Switch back to the Geometric Objects tab of the Select window and make sure that root is selected. Go back to Post Particle Effects tab and drag & drop the smokeparticle effect to the view window. This applies the post particle effect to the currently selected level (which is the root level in our example).

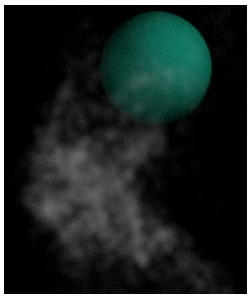


The hierarchy of the completed scene

The scene is now ready. From the Geometric Objects tab check that your object hierarchy is similar to the example image.

Having an other object than root selected while dropping the smokeparticle effect to the view window adds a new hierarchy level (which contains the selected object and the mapping). This would confuse the simulation preparations we already made.

15. You may test render the view window to see how it looks like. Adjust the Particle Size of the effect and/or the Fade value of the particle object if you need to change the thickness of the smoke.



16. Open the animation window (from the View menu) and set the frame count to a suitable value (200 - 300). Play the animation to see how the particles behave. If they are not moving fast enough, decrease the mass of particles. Another important attribute for controlling the velocity of particles is the Fluid Velocity attribute of the fan (in the Phys tab of the property window). It defines the terminal velocity of the particles.

The final step is to render the whole animation. See the rendering tutorials if you are not yet familiar with file rendering. Before starting rendering, make sure that Default Effects (or any other post processing configuration you plan to use) includes the Render Particles module.

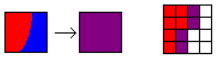
Compositing

The post processing system has an important role when preparing rendered images for a compositing system (software/hardware that reads image sequences and blends them together using alpha channel and other information). Especially the VSL effect is useful in this context. The user can define any necessary channel modification using it.

In this section, three compositing cases are examined:

- A post processing configuration that prepares images for alpha channel keying (over a background video, etc.)
- A post processing configuration which reads an image sequence from a disk file and does the alpha preparation as a separate rendering pass
- A post processing configuration that keys an image sequence over another image sequence

Tutorial level: Medium/advanced



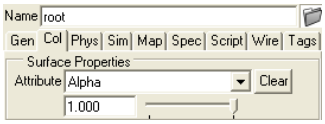
High quality rendering requires use of anti-aliasing. In a compositing situation, anti-aliasing has one undesirable side effect: The edge pixels of objects become blended with the background color. For example, if you render the scene over a blue background, some pixels will inevitably include color both from the rendered object and from the background.

The leftmost square is a magnification of a pixel, which is split by an edge of a red object. After rendering, the color of the pixel is the average of all colors included in the pixel - dark purple shown in the second square. If the rendered scene is then 'superimposed' over a new background, the purple pixels become an undesirable glow or color error. The third square representing a 4*4 pixel array shows this situation on a white background.

A suitable VSL effect can remove this problem. The next example demonstrates this.

Example file: 'tutorprojects/postprocessing/nobackgroundantialiasing'.

1. Model a simple test scene: add a couple of objects with different colors and add a light source or two.

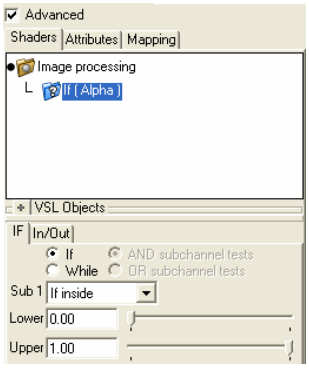


Alpha=1.0 makes objects non-transparent in compositing

2. Compositing usually requires use of the alpha channel. Select the 'root' level of the scene, open the property window and go to the 'Col' tab. The 'Surface Properties' frame includes 'Attribute' list - select 'Alpha' from it and set the value slider below to one. In most compositing systems alpha level one means full non-transparency. If the scene contains glass (or otherwise transparent) objects, you set an alpha value of 0.5 - 0.8 for them.

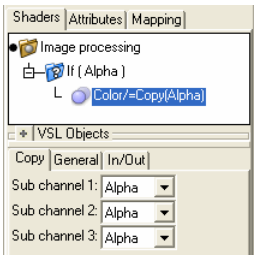
3. Go to the Post Image Effects tab of the select window. Select 'New/Post Image' from the popup menu. Then select the 'New/VSL effect from the popup. Two new objects appear to the end of effect list.

4. The property window shows the latest created effect - the VSL effect. Change its name to a more suitable one, 'removebackgroundaa'. The effect is currently empty and does nothing. Check the Advanced option to open the VSL editor, and open the VSL Objects frame.



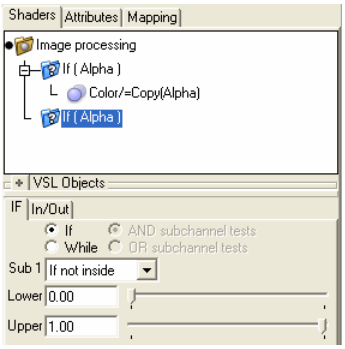
5. Drag and drop an If object from the VSL Objects into the Image processing node. Open the popup menu of the VSL hierarchy gadget and set Input 0 to Image:Alpha. The lower part of the property window shows If specific properties. Select If Inside from the gadget labeled with Sub 1. Set Lower to 0.001 and Upper to 1.001. Now the If object tests if the Alpha channel value is non-zero - in other words, if the pixel is not a purely background colored pixel. Other sub channels labeled as Sub 2 and Sub 3 need not to be tested, because the alpha channel has no sub channels (unlike the Color channel with R, G and B sub channels).

6. Drag and drop a 'Copy' VSL object into the 'If' node of the VSL tree. Using the popup menu, set 'Source' to 'Image:Alpha' and 'Operator' to 'Divide'. Set all the three sub channel selectors on the bottom of the property window to 'Alpha'.



Now the VSL code divides color of all pixels that have non-zero alpha level with the alpha level. This compensates the blending with a black background, because alpha measures the amount of non-transparency included in a pixel.

There may be some isolated pixels where the alpha level is very small ('If' operator accepted alpha level greater than 0.001). We may clean these up as follows:

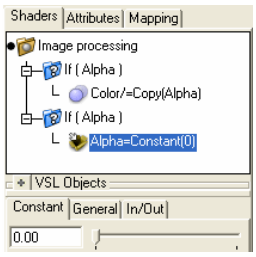


7. Drop a new 'If' object into the 'Image processing' node. Set exactly the same 'Input 0', 'Lower' and 'Upper' as in the step 5 above, but this time select 'If not inside' to the 'Sub 1' sub channel test selector.

8. Drop a 'Constant' object into the new 'If' object. Select 'Output' = 'Image:Alpha' from the popup menu. In other words, we reset very low alpha channel values to zero. This is in most cases only a 'cosmetic' operation, because alpha values below 0.001 are truncated to zero by all common file formats (such as Targa). The VSL effect is now ready.

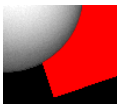
9. Select the 'Post Image' object which was created in the beginning of the tutorial, from the select window. We configure its contents next using the property window. If the scene contains particles, drag a 'ParticleEffect' object from the 'Available Effects' to the 'Active Effects' list. Then drag & drop the created VSL effect to the 'Active Effects' list. Finally rename the post image in a suitable way, for example 'No-background-aa'. Close the property window.

10. The post processing configuration is now ready for use. We can try it by rendering the view window. Drag and drop the post image (No-background-aa defined above) from the select window to the view window. Then start ray tracing.



The final structure of the VSL effect

You will see the usual shaded rendering appearing to the window, with anti-aliasing that makes object edges appear smooth and nice. After a while, the post processing replaces it with a new image where edges against the background look jagged and ugly! Do not worry, this is exactly the correct result. The alpha channel contains the information for making the edges smooth again when the image is merged over a new background.



A part of rendering magnified. Note, how interior pixels are nicely anti-aliased, but pixels against the background are not



Note

The above mentioned post processing configuration works only when rendering over a black background. Do not use a backdrop image with it.

Preparing images for compositing using a second rendering pass

When using motion blur, the background anti-aliasing elimination presented above cannot be used. The motion blur system assembles images together after post processing and the background color becomes blended with the object colors again. It is better to render the animation frames in the usual way and make another rendering pass to fix the background anti-aliasing.

Example project: 'tutorprojects/postprocessing/preparecompositing'.

1. Load in the scene modeled in the previous tutorial ('tutorprojects/postprocessing/nobackgroundantialiasing'), unless you continue directly from it.

2. We will need a short and simple test animation. Open the animation window (from 'View/AnimationWindow' menu) and set frame count to, say, 5 frames only. Select one of the scene objects, move the time slider to the last frame, activate animation recording button and move the object to a new place. Turn recording off and rewind the animation.

3. Prepare motion blur rendering (see the rendering tutorial 'Motion Blur' for details): Activate 'Motion Blur' using the 'Windows/Animation Settings' menu. Four samples are enough for the basic test. Set anti-aliasing of 'Reasonable Quality' rendering settings to 'Stochastic' and anti-aliasing level to zero. Make sure that the Output objects tab of the select window includes a 'Targa RGBA' object - if not, create one. Add a camera object to the scene.

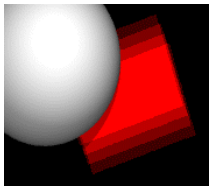
4. Open the file rendering window. Define a suitable file name (e.g. 'aa_test'). Add 'Targa RGBA' to 'Active File Formats' and remove possible other active formats. Select 'Reasonable Quality' to the 'Rendering Settings' gadget. Set 'Image Effect' to 'Default Effects' or leave it empty.



Note

Do not select the No-background-AA configuration yet. Set the resolution to a suitable test value, say 300*300 pixels. Hit the 'Render Animation' button and wait until rendering is finished.

So far we have not done anything new - the above mentioned briefly described steps only render a motion blurred test animation to a sequence of Targa files. If you take a look at one of the rendered images, it looks as in the example image on the right.



The image shows that the red moving rectangle is blurred with the black background. This will be a serious problem if the sphere and rectangle are composited over a white background. The second rendering pass fixes the Targa image sequence. This is the actual issue of this tutorial and is described next.

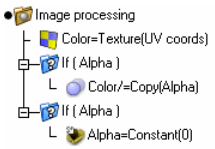
5. The scene is no longer needed, so delete all objects. Open the Animation Settings window from the 'Windows' menu and turn off 'Motion Blur'. The second rendering pass manipulates already blurred images and the motion blur should be switched off this time. Close the Animation Settings window.

6. Go to the rendering settings tab and create a new settings object. Open the property window to edit the rendering settings. Rename it as 'Compositing settings'. The purpose is to prepare a rendering settings object, which renders the empty scene as quickly as possible - the actual work is done in post processing. Set 'Undersampling/Horizontal & Vertical' both to the level 4 and clear the 'Interpolate' option. Turn off 'Box rendering' from the 'Distrib' tab. You should not use network rendering during the second rendering pass, because the simple configuration is probably faster to compute locally instead of sending image data across the network.

7. Switch to the Post Image Effects tab of the Select window, and select the VSL effect 'removebackgroundaa'. This time it should not process ray traced data (from the empty scene), but Targa disk files. Therefore, we have to modify the effect a bit. So, open the property window (if not open yet).

8. Check the 'Advanced' options of the property window if not checked. The VSL effect already contains the two 'If' blocks which were inserted in the previous tutorial. Drag a 'Texture' object from 'VSL objects', move the mouse pointer over the top edge of the upper 'If' icon before releasing the left mouse button (the cursor changes to an up arrow when insertion point is correct) and drop the texture object there. The texture should be read before 'if' tests, therefore it should be first object in the VSL tree.

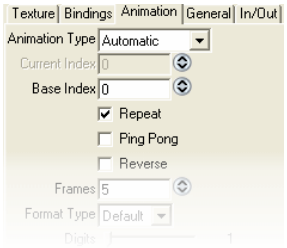
9. At the top of the property window you can see a gadget labeled as 'Image Coordinate Channel' and it is set to 'UV coords'. Therefore, using the popup menu of the VSL tree gadget, set 'Input 0' of the inserted texture object to 'UV coords'.



Color channel loader inserted

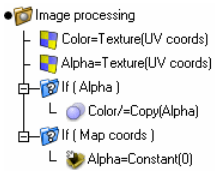
10. The inserted Texture object should load images of the motion blurred Targa sequence. Therefore, we have to define properties of an animated texture. Select the Animation tab from the bottom of the property window. Set Animation Type to Automatic for automatic frame indexing.

11. Switch to the Texture tab and enter the name of the rendered images without index but with the format postfix to the File Name gadget. For example 'AA_test.tga'.



Options of an animated texture

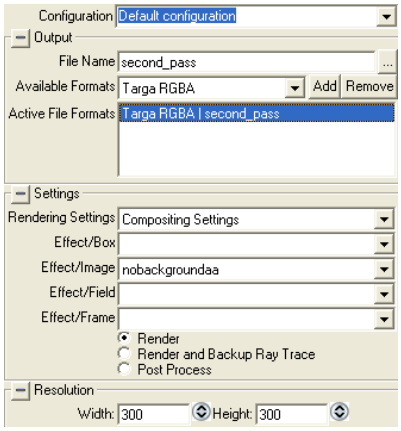
12. Make sure that the Texture object is still selected and choose Copy from the popup menu of the VSL tree gadget. Then select Paste from the same menu. A copy of the Texture object appears to the end of the VSL tree. Drag and drop it just below the first Texture object (above the If objects). Change the Output to Image:Alpha using the popup menu. Close the property window.



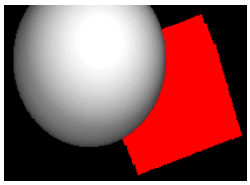
The finished VSL structure

The VSL code for the effect is now ready. Instead of processing rendered image data, the effect removes background anti-aliasing from a sequence of disk files.

13. Open the file rendering window. Change the file name to a new one, e.g. 'secondpass'. The active output list should again include only 'Targa RGBA'. Select the fast-to-render 'Compositing Settings' to the 'Rendering Settings' gadget. Set image effect to Nobackgroundaa (=the post image including the prepared VSL effect). Use the same resolution as in the first motion blur rendering. Hit Render Animation to start rendering.



File rendering settings for the second rendering pass



Background color blending eliminated

Attached is a processed sample image. There is no motion blurring left at all and the anti-aliasing is poor. Alpha channel includes the necessary information for blending the image over a new background, as the next tutorial shows.

Compositing using the alpha channel

The Targa image sequence, which was processed in the previous tutorial, includes alpha channel information and is suitable for use in various composition systems. For example, a video hardware device can insert the computer generated graphics over a live video signal.

It is also possible to use Realsoft 3D itself for doing the compositing job. Thanks to VSL, the user can freely define any kind of keying algorithm, which blends two or more image sequences. The next example shows how to achieve this.

Example file: 'tutorprojects\postprocessing\compositing'.

1. Go to the Post Image Effects tab of the Select window. Select 'New/Post Image' from the popup. Then select 'New/VSL Effect' from the popup.

2. Open the property window. Check the 'Advanced' button to edit the new empty VSL effect. Rename the effect as 'Compositing by VSL'.

3. Drag and drop three 'Variable' objects from 'VSL Objects' frame into the 'Image processing' node of the VSL tree. Set the names and types of variables using the controls at the bottom of the property window as follows:

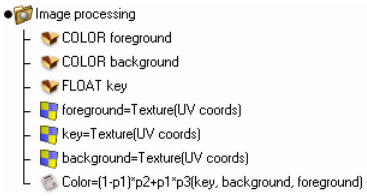
- Variable 1: name = foreground, type = Color
- Variable 2: name = background, type = Color
- Variable 3: name = key, type = Float

4. Drag and drop a 'Texture' object to the VSL tree. Set 'Input 0' = 'Image:UV Coords' and 'Output' = 'foreground'. Define the properties of the animated texture sequence exactly as in the previous tutorial, but this time use the second, processed sequence: ('Animation Type' = "Automatic', 'Frames' = 5, 'File Name' = 'secondpass.tga').

5. Copy and paste the texture object. Change 'Output' of the pasted texture object to 'key' using the popup menu.

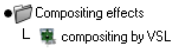
6. Drag and drop a third 'Texture' object to the end of the VSL tree. Set 'Input 0' to 'Image:UV Coords' and 'Output' to background. Select a suitable background image or image sequence. For example, use the good old 'textures/lake.jpg' image.

7. Now we have inserted three texture loaders. The remaining step is to combine their information. Drop an 'Operation' object to the end of the VSL tree. Set the 'Type' of operation (using the gadget at the bottom of the property window) to the long formula at the end of type list: '(1-p1)*p2 + p1*p3'. Mathematically oriented users may notice that this is the formula of linear interpolation. Using the popup menu, set 'Input 0' to 'key', 'Input 1' to background and 'Input 2' to 'foreground'. The default output is already 'Image:Color' which is the correct target channel.

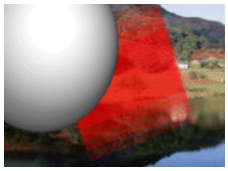


The VSL effect for image composition

8. The VSL effect is now ready. Create a new post image effect object from the select window. Rename it as Compositing effects. Insert only one effect to it, namely Compositing by VSL. Close the property window.



9. Open the file rendering window. Define a new file name, for example 'thirdpass'. The output can again be a Targa file, but you can also use other formats such as 'Windows BMP'. The VSL effect above does not maintain the alpha channel (because it is already used once for compositing). Set 'Compositing settings' to the 'Rendering Settings' gadget and 'Compositing effects' to the 'Image Effect' gadget. The resolution should be the same as in the previous rendering passes. Start file rendering by hitting the 'Render Animation' button.



The sample image above shows how the blurring appears again, but now with a new background photograph. Smooth anti-aliasing is included.

Summary

The detailed tutorial examples as outlined above, constructed all the necessary compositing tools and objects from the very beginning. In a real situation, you can reuse these example projects:

- Render the animation to targa files in a normal way.
- Load Rendering Settings and Image Effects sections from the 'preparecompositing' sample project. Edit the Texture objects of the VSL effect to include correct file names and frame counts. Render the animation second time using the same resolution, quick compositing optimized rendering settings and the loaded post processing configuration.
- Load Image Effects sections from the 'compositing' sample project. Edit the Texture objects of the VSL effect to include correct file names and frame counts. Render the animation a third time using the same resolution, the quick compositing optimized rendering settings and the loaded post processing configuration.

Of course, you can include the necessary post processing, rendering settings and file rendering configurations to your startup project. Then all the necessary tools would be available all the time.

Animated Lens Flare

Particle effect properties, such as the rotation angle of lens flare streaks, can be animated easily using key framing: simply enter a suitable attribute value on the property window, with animation recording turned on. In this example, we show how to set up a more complicated system: the rotation angle of flares based on the cameras orientation.

Tutorial level: Advanced

Example project: 'tutorprojects/postprocessing/rotatingflares'

1. First create the camera: activate the camera tool and hit Accept to record the current view to the camera. Turn animation recording on and go to the last frame of the animation. Open the camera's property window, go to the Gen tab and change the Rotation fields first component by say 180 degrees (for example from zero to 180 degrees). Then turn animation recording off. Camera heading will now change during the animation half a revolution.

2. Activate the particle tool and create some random 1D particles, then hit Accept button to finish the tool. Open the property window, go to Spec tab and turn off the Rendering/Scanline option. Particles will be rendered using only post effect flares.

3. Go to the Post particle effects tab of the Select window. Drag&drop the Stars effect (or another lens flare) to the view to map it to selected particles. Render the view to make sure that flares work as expected (if not, make sure that the view window's post effect configuration includes particle effect rendering).





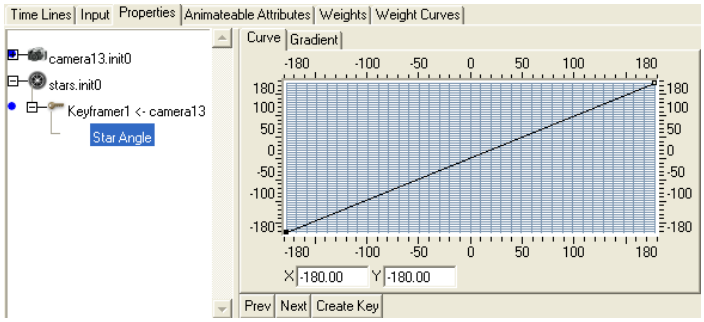
4. While the Stars effect is still selected, right click and pick the popup menu Make Choreographable on the select window. Next open the Choreographs window from the Animation pulldown menu.

5. On the Choreography window, select the stars:init0 item. Select New/Keyframer from the popup menu. Double click the stars:init0 item to open it, and select the keyframer1 object under it. Switch to the Input tab. Then go to the Select window, switch to the object geometry tab and select the camera object. On the choreography window, set Input=Geometric Object and click the Change button. The Input area is changed to show camera attributes. Select Rotate Quaternion.x from the attribute list - in other words, the cameras heading angle will act as the input for the flare animation.

6. Switch to the Time lines tab of the choreography window. Make sure that the keyframer1 item is still selected, and define a time line Start=-180, End=180 using the numeric fields on the bottom of the window. This timeline covers one cycle of the input parameter (=camera's heading angle). To ensure that all heading angle values become mapped to the range where the animation curve is defined, set the Periodic option at the bottom of the choreography window.

7. Next we define what to animate. Switch to the Animateable Attributes tab and pick Star Angle from the list. Check the Animated option above the list.

8. Finally go to the Properties tab of the choreography window. Activate the Star Angle item under the Keyframer1 choreography. Then move the mouse over the curve gadget and select Create predefined curve/45 degree. curve from its popup menu. A diagonal line appears; activate the start point and change its coordinates to X=-180, Y=-180 degrees. Activate the end point and change it to X=180, Y=180 degrees. In other words, the curve maps the cameras heading angle to the identical lens flare streak angle. Other kinds of curves could be used as well, of course.



The animation is now ready. Render it to an AVI file to see rotating streaks.

Post Shading

Tutorial level: Advanced

Example project: 'samples/rendering/post shading studio'.

In this tutorial we will examine how to collect channel information from the ray tracer in order to evaluate the basic illumination model in the post processing phase. Post processed surface shading has some advantages:

- Sampled illumination data can be filtered before the shading. This reduces noise.
- Recomputing the post shading after illumination parameter adjustments is often quicker than rendering the image from scratch. The speed advantage is remarkable when the scene geometry is complex and slow to ray trace.

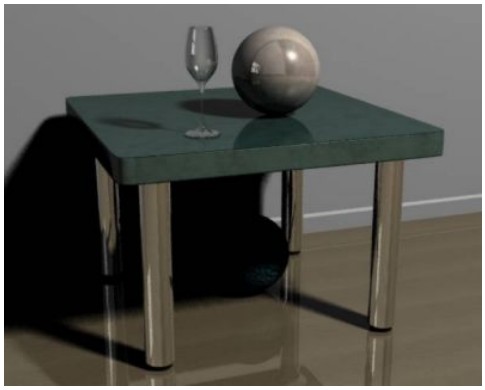
The illumination model of Realsoft 3D - and the rendering pipeline in general - is open and customizable. This means that the parameters of a post shading system are not fixed. The user can any time code a new VSL material, which changes surface illumination in a totally new way. It's impossible to reproduce the functionality of the new shader accurately in a post shading system, which assumes other kind of shading behavior.

A simple post shading system renders very quickly and allows interactive adjusting of parameters. A complex post shading solution which controls the result image in a detailed manner - providing even object or light source specific parameters - is slower to use, but naturally provides much more possibilities to adjust the image. The optimal choice between speed and flexibility depends on each scene.

The standard setup of Realsoft 3D includes a set of example components for post shading. These components allow quite a detailed control over post shading. The components are built using the standard VSL objects and post effects, and therefore they can be modified and tailored to suit all kind of projects. For example, unused elements can be deleted to speed up rendering.

Let's experiment how to use the post shading system, which is included in the program setup.

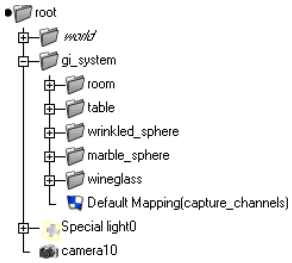
1. Model first a test scene: a floor and a wall, a table, a glass object, a reflective object, etc. Feel free to use your imagination. Add a large sphere (sky dome) around the scene. Open the property window, go to the Col tab, select Surface Properties/Attribute = Illumination and enter a value 0.5 0.5 0.5. This makes the sky dome self-illuminated. Collect all elements of the environment (the sky dome etc.) under a hierarchy level 'world' and all directly visible surfaces of the room and the furniture under another level 'gi_system'. We will later apply post shading only to the items which are directly visible in the image; this saves some rendering time.



The test scene. Shadows are noisy because of minimal area light sampling.

Add also a rectangular special light source into the room. You can leve the Quality setting of the area light source to the minimal value 0. If you test render an image, shadows will show clearly visible sampling noise. Turn also view's flashlight off, because the post shading system includes its own independent flashlight.

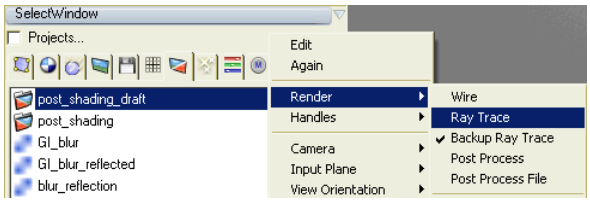
2. Go to the material tab of the select window. Choose New/From Template from the popup menu of the select window. Use the opened file browser to find and load in the material 'realsoft3d/materials/illumination/capture channels'. Select the level 'gi_system' and drag and drop the loaded material 'capture_channels' to the view window. This maps the shader to the selected level. Shading information will be collected only from the objects inluuded in the 'gi_system' level.



The channel capture shader mapped to gi_system hierarchy level

3. Switch to the render settings tab of the select window. Drag and drop the render settings object 'GI_rendering' to the view window. GI_rendering does not compute anti-aliasing, which is usually quite important. The post shading system cannot afterwards separate illumination components from antialiased pixels, and therefore ray tracer antialiasing would cause shading errors. Antialiasing must be done after the post processing by down scaling the image resolution.

4. Go to the post image effects tab of the select window. Open the popup menu and choose Paste from a File. Load in the file 'realsoft3d/posteffects/post_shading'. Drag and drop either post_shading or post_shading_draft from the select window into the view window. The draft version does not use supersampled resolution to antialiase and is therefore 4 times faster.



Set image effects to post_shading_draft using drag and drop, activate Backup Ray Trace and Render

5. Open the popup menu of the view window. Make sure that Backup Ray Trace option is enabled. Then select Ray Trace.

First the ray traced image appears, then, after a while, a post-shaded version. The post shaded version is brighter, because it adds indirect illumination to the image. The dithered shadows are now smooth.

The next step is to experiment with the post shading parameters. Switch to the post image effects tab of the select window, and open the property window.



Shadow quality improved

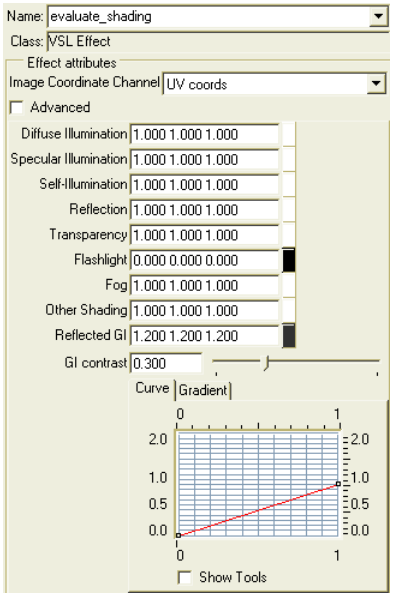
6. To make reflections softer, select the blur_reflection effect and increase the Blur Level value using the property window. Then select Render/Post Process from the view popup menu. A new image appears much quicker, because ray tracing was not computed.

7. To make shadows even smoother, select the blur_shadows effect. Increase either the Blur Level value or the Iterations value. Then apply Render/Post Process.

8. To remove all reflections from surfaces, select the evaluate_shading effect and set Reflection to black (0,0,0). Then apply Render/Post Process.

9. Change the Flashlight value of the evaluate_shading effect to pale yellow (0.4, 0.4, 0.1). Post process again. The image gets a warm fill light effect

10. Set Specularity to (1.6 1.1 1.1), then Post Process. Specular highlights become brighter and get a red tone around the white brightness maximum.



Post shading parameters



Note

You can view the source data, from which the image is computed, using the channel menus located at the end of the view's Render popup menu. It is often useful to examine the contents of the deep backup image.

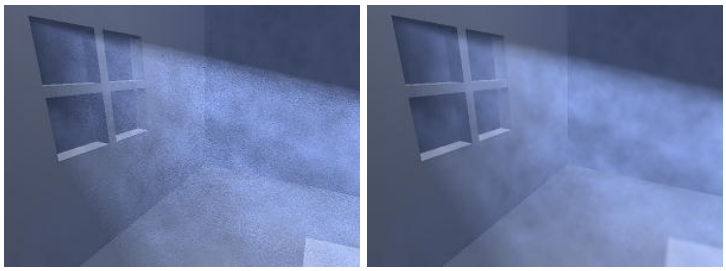
As the experiments above show, the most important component of the system is the evaluate_shading effect. It is a VSL effect, which weights the stored channels and then recombines them to a new image. The VSL structure of the effect is quite straightforward and easy to customize. By default, all weights are set to the level (1,1,1), which reproduces the original ray traced illumination. Note, that although the weights are presented as RGB colors, their range is not limited to 0..1 interval. The full list of shading parameters is explained below.

- Diffuse Illumination: rescales the usual diffuse shading. This is the most common shading component.
- Specular Illumination: the weight for all non-diffuse illumination components from light sources. Specular highlights, ambient lighting and user defined Surface Illumination shaders contribute trough this parameter.

- Self-Illumination: controls self-illumination of surfaces (usually defined as Surface:Illumination = RGB).
- Reflection: controls mirror reflections.
- Transparency: controls optical transparency i.e. refracted rays (not faded surfaces).
- Flashlight: the fill light from the camera. Replaces (not rescales) the camera flash light.
- Fog: the weight for a global fog surrounding the camera.
- Other Shading: unusual custom effects which are not included in other post shading parameters.
- Reflected GI: global illumination via reflective and transparent objects.
- GI contrast: enhances bump maps and other fine details of surfaces.
- GI control curve: the gamma correction curve for global illumination.

It should be noted that these parameters control only the first step of the recursive ray tracing. For the sake of speed and memory consumption, the system stores the channel values only from the first step of the ray trace recursion. For example, a self-illuminated surface which is seen in a mirror reflection does not become darker when you decrease the self-illumination parameter. Instead, the reflection weight controls its appearance.

The fog blurring effect, which is part of the post shading system, needs some special attention. With it, you can reduce the sampling noise of volume sampled fog shaders. Applying a pixel accurate fog shader to blurred shading components sometimes creates clearly visible errors. You can compensate this by making fog blurring as large as the maximum of the other blur components, such as the shadow blur effect.

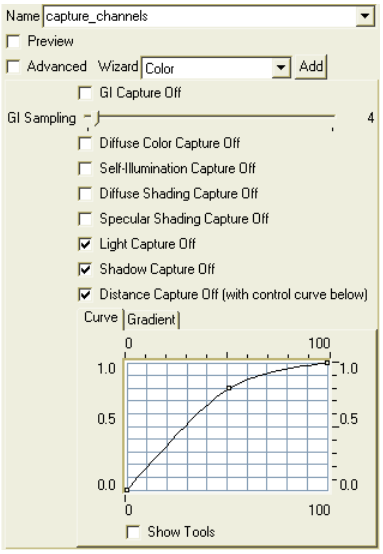


Foggy atmosphere before and after post effect blurring

Let's take a closer look at the channel capture material. Go to the material tab of the select window, make sure that the material capture_channels is selected and open the property window. The material interface displays a large number of switches 'XXX Capture Off'. With these switches, you can turn off collection of unused data and consequently speed up rendering. By default, all channels used by the post effect shading are enabled. Light, Shadow and Distance capture are disabled, because they capture additional channel data which is not needed in post shading.

This material can be used in collecting information for all kinds of post processing purposes. The switches and their relation to underlying channels are:

- GI Capture Off: do not capture global immumination.
- GI Sampling: number of GI rays (when GI capture is enabled).
- Diffuse Color Capture Off: do not capture diffuse color to diffuse_color channel.
- Self-Illumination Capture Off: do not capture self-illumination of surfaces to self_illumination channel.
- Diffuse Shading Capture Off: do not capture diffuse shading by light sources to diffuse_shading channel.
- Specular Shading Capture Off: do not capture non-diffuse shading by light sources (specular, ambient light, user defined illumination shaders) to specular_shading channel.
- Light Capture Off: do not capture the unshadowed light from light sources to light channel.
- Shadow Capture Off: do not capture the amount of shadow to shadow channel.
- Distance Capture Off: do not capture distance (z buffer) to depth channel.



The channel capture controls

Some important notes:

- Specular Shading Capture collects all non-diffuse illumination elements. For example, ambient lights are included here. The name is chosen like this because specular highlights are probably the most common illumination effect in this category.
- By enabling Shadow Capture, you can output the shadow channel into a shadow mask file. The shadow channel is not used by the post shading system.
- Distance Capture computes a non-linear curve controlled z buffer value to the depth channel. Distances from 0 to infinity are mapped to the 0..1 interval, which means that the z buffer does not overflow or truncate the values regardless of the used datatype (byte, word, float etc.). The depth channel is not used by the post shading system.

The idea behind the capture_channels material is that in includes all surface related shaders. In each shader, the material examines how the total illumination and other relevant channels change and stores the observed changes to a set of additional channels diffuse, ambient, etc. As explained earlier in this tutorial, only the first step of ray trace recursion is stored.

Technical Outline Drawing Style

A suitable post effect configuration can extract object outlines and control the line width to achieve a drawing style, which is familiar from technical illustrations.

Tutorial level: Advanced

Example project: 'samples/rendering/outline_drawing'

1. Load in 'outline_drawing.r3d'. To apply it to your existing geometric objects, enable 'Selected Sections Only' in the File Load window and uncheck 'Objects' section. Use 'Replace' loading if you want to clean up some unnecessary items. Otherwise clear the Replace option to add the items to your current libraries.

2. If the scene is not included in a closed environment, add a camera backdrop to the current camera. Otherwise empty space areas will render fully black.



Note

A backdrop plane works well in orthographic rendering. If you plan to render a perspective image, create a camera-centered analytic sphere around the scene. In orthographic rendering, backdrop plane has a constant distance from the camera. In perspective rendering, a sphere defines the constant distance background.

3. For maximal rendering speed, you can delete all materials except perhaps some outline area definitions, which you might use, from to the material library. The example configuration outputs only black and white lines so material evaluation will not be needed. Do not delete materials if you want to customize the style and include your own effects, which use the shaded color information.

4. Go to render settings tab and drag and drop 'Outline_render' to a view window. If maximal rendering speed is important, you can raise the Undersampling x and Y values from 1*1 to a higher level. Especially when rendering a high resolution image to a disk file, undersampling can be raised up to the maximal level.

The characteristic features of the Outline_render settings are:

- Distance and bump normal channels trigger more accurate sampling
- Lighting and volumetric computations are turned off and antialiasing is zero
- Antialiasing method is geometric to keep sampling intervals regular

5. Go to the Post Effects tab of the select window and drag&drop TechStyleLines to the view window. Check in the view property window that you do not have other 'per image' post effects applied.

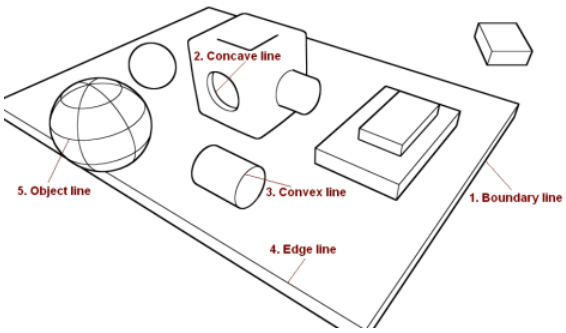
6. Render a test image. If you get too many lines, select the Edge Detection post effect and increase the Threshold and Normal Threshold values. Lowering the thresholds adds more lines to the image. If some object borders are drawn with a thin line although they should be thick, select the object_boundary_lines effect and lower its threshold value.

7. 'draw_lines' effect defines the line width range from thin to thick lines. Change it if necessary.

8. If you do not want to see all surface and object borders, select the post effect include_object_outlines and set the 'Exclude object outlines' checkbox. Note that you can customize the object outline density using materials which assign Surface:Identifier channel values. This example project includes 12 such materials. For example, the hierarchy below forces object outlines between part1 and the rest of the scene. There will be no other object boundaries.

9. 'Linewidth control' effect defines line thickness categories. Currently 5 line types are available:

- 1. Boundary line width = Width of lines between 2 disconnected surfaces
- 2. Concave line width = Width of lines between connected surfaces with a concave angle
- 3. Convex line width = Width of lines between connected surfaces with a convex angle
- 4. Edge Line Width = Base width of all lines
- 5. Object line width = Width of lines between separate geometric (connected or disconnected) objects and surfaces. This Division can be disabled or customized using materials.





Note

These lines types are not exclusive. If a line meets two or more type conditions, maximum width is used.

10. For the best possible quality, select the TechStyleLines effect, open the property window, go to Scaling tab and set Output Scale X and Y to 0.1. A lower value such as 0.25 is suitable for test renders.



Note

Edge detection depends on the image resolution and post image's Output Scale value. High resolution means that trigger channels usually change less between adjacent pixels. It may be necessary to readjust the edge detection thresholds after selecting the final output resolution.



Note

The post effect combination uses many channels, some in floating point form. The deep image requires very much ram. Use of a high number of render boxes (Render settings/Distr tab) is therefore recommended.



Chapter 10. Animations

Getting Started with Animation

Animation is the process of specifying how objects act over time. There are several ways to animate objects in Realsoft 3D. The most common form of animation is called keyframing, in which the positions, shapes, and numerous other attributes of objects are set at important (key) times. It is then left to the computer (or in the old days to junior artists) to provide the in-between frames, often called "tweens". The number of keyframes you must set usually depends on how precisely you need to control the objects in a scene. Realsoft 3D provides a number of tools to help you achieve exactly what you envision.

Before proceeding with the animation tutorials, make sure the Animation window is open. You can open this window by selecting View/Animation Window from the main menu bar. Note that the animation tutorials assume that you already know how to perform tasks like creating geometric objects such as spheres.
Tutorial project:'tutorprojects\animation\gettingstarted\moving sphere'

Tutorial level: Beginner

Simply stated, keyframing defines the starting and ending values of an object attribute, such as its position. Think about causing an object, such as a sphere, to move in a straight line. There is a starting position and an ending position. Nothing else needs to be specified. However, what if you want the sphere to move along a curved line? Clearly, you can no longer just specify a start and an end position. You need more positions to define the path.

To set keyframes in Realsoft 3D, you make modifications to an object while in animation recording mode. Let's begin by creating a simple animation of a sphere moving in a straight line.

1. Create an analytic sphere near the left top corner of the view window.

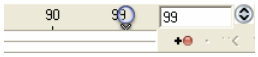
2. Click the red Record button in the animation window.



Click the red Record button to activate animation recording

3. Move the time slider to the last frame.

4. Move the sphere near the right bottom corner of the view window.



Move to the last frame using the time slider

Our moving sphere animation is now ready. Click the Record button again to exit animation recording mode and then click the Play button in the animation window to preview the animation.

The sphere moves in a straight line from the top left of the view to the bottom right. Pretty straightforward so far. So what about making the sphere move in a curve from top left to bottom right?

5. Move the time slider to frame 75. Note that as you do so, the sphere moves according to its currently defined animation.

6. Turn on animation record, and move the sphere to the bottom left of the view.

7. Exit animation recording mode.

Now click the Play button. The sphere now moves along a curved path from top left to bottom left to bottom right. As you can see, by setting more keyframes, you gain more precise control over an object's animation.

Keep the current project active. We'll use it in the next section.

Animating the color of the sphere

Tutorial project: 'tutorprojects\animation\gettingstarted\moving sphere with animated color'

Tutorial level: Beginner

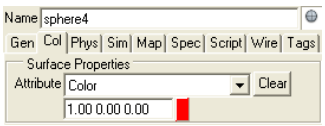
So, moving the sphere was a simple exercise in specifying where you wanted it to move from and where you wanted it to move to in the view window. But, it was mentioned earlier that all types of object attributes can be animated over time. Obviously, all of an object's attributes cannot be modified through a view window. For instance, what if you want to have a chameleon sphere that changed color over time? Color is not an attribute you can control through the view window. Any object attribute can be made to change in animation recording mode by changing its value through any of the available interfaces in Realsoft 3D, such as the object property window.

Let's make our moving sphere change to red in the frame 40 and blue in frame 100.

1. Activate animation recording mode and set the current time to frame 40.

2. Use the property window's Color tab to change the color of the sphere to red.

3. Now change the current time to frame 100, and set the color of the object to blue.



Modify the color of the sphere in the property window

Exit animation recording mode and view how the color of the sphere changes while you move the time slider back and forth along the timeline.

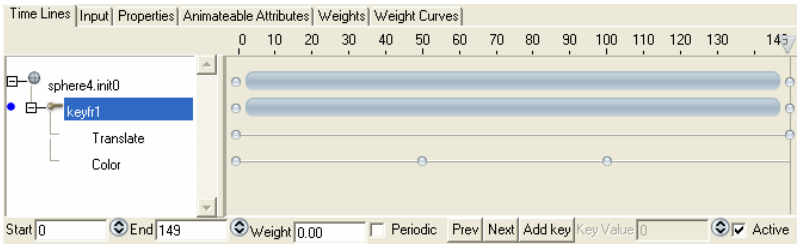
Editing key frames

So, now you have a sphere that moves and changes color. Assume that the movement and colors are okay, but the timing of the color change is not what you wanted-the color change takes too long and occurs too soon. How can you make such changes without re-recording the animation? An object's animated attributes can be edited using the Choreographer.

1. Select Windows/Choreography Window from the main menu bar.

The choreography window is opened. It shows one choreography named sphere.init0 (or something similar to this name).

2. Go to the Time Lines tab of the choreography window. Double click the 'sphere.init0' choreography object to open its sub structure. It contains one item, 'keyfr1'. Double click it, too.

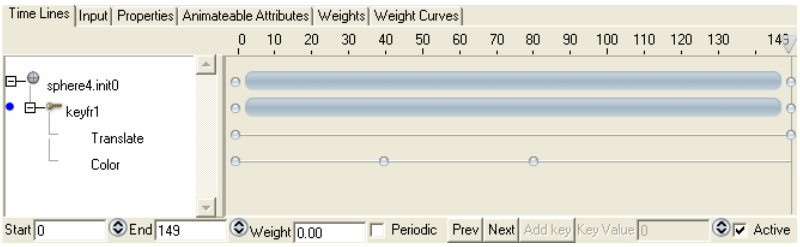


Time lines of the animated sphere

You can now see two new names below the 'keyfr1' choreography indicating that the choreography animates two attributes: Translation and Color. Each of these two attributes has its own timeline. A key indicator is shown as a small rectangle on the timeline.

To move a keyframe, drag its key indicator with the left mouse button. Along the timeline of the Color attribute, move the key at frame 40 to the right to frame 50, and then move the key at frame 100 down to frame 80.

To position a keyframe precisely, you can select a key indicator by clicking it with the LMB, and then enter a new frame number in the Key Value field at the bottom of the window.



Sphere turns to red in the frame 50 and to blue in the frame 80

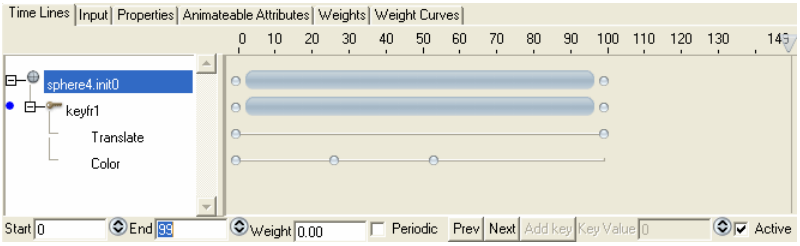
If you now move the time slider, you can verify that the color of the sphere really turns to red in the frame 50 and to blue in the frame 80.

Scaling time lines

Let's now say that you like the overall movement and color change of the sphere, but that the overall timing is too long. Rather than 150 frames, you need the animation to play in 90 frames. This is easily accomplished using the choreography window. If you adjust the end knob of the 'sphere.init0' time line, the result is that all animation components of the sphere object are modified. All the keyframes are scaled accordingly.

To change the length of the 'sphere.init0' time line, do the following:

1. Drag the ending time indicator of the 'sphere.init0' timeline from frame 149 to frame 90 (or select the choreography and use the Start and End fields to define the timeline numerically).



Sphere.chor0 choreography is changed to execute within frames 0 ... 90

If you now play the animation, the sphere moves in a curve to the bottom right corner of the view window during the frames 0 to 90 and then stops.



Note

The time line window can be zoomed and panned like many other windows in the program. To pan, press Ctrl and drag with RMB. To zoom, press Shift and drag with RMB. You can also adjust the vertical splitter control between the choreography name list and the time line canvas.

In addition to this, there are couple of other modifiers specific to the time line window:

- *Alt* + *drag* a time line modifies only the time line of the choreography without scaling the actual keys with the time line.
- *Shift* + *drag* snaps the selected knot to the nearest frame.

Animation Controls

Besides creating an animation, there are a number of *controls* for the animation system that you need to set, such as the length, and the type of time code and video system to use. This tutorial shows you the animation controls that are available in Realsoft 3D.

Animation length

You can change the total length of your animation at any time using the frame count field in the animation window.



Animation consisting of 150 frames

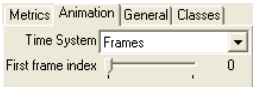
To add new frames to your animation (increase the length), enter the desired value in the field and press the Enter key. A dialog appears asking if you want to scale the timelines for any animated objects.

To add new frames without affecting the length of any currently defined animations, select No (do not scale the time lines). If you want to scale the existing animation timelines to include the new frames being added, select Yes.

You remove frames (shorten) from an animation by the same process as adding frames: type a new length into the frame count field and press Enter. However, be aware that if you decrease the total number of frames to less than the length of any particular choreograph, and do not scale the timelines, such choreographs will be affected. Again, you always have the option of using the Time Lines tab of the choreography editor to make adjustments.

Setting the time code and video system

Before you start an animation project, you should set the type of time system and video system to use. To access the Time code settings, open the Preferences window from the main menu



Selecting time units

File # Preferences # Window

and click the Animation tab.

For the Time System, you have three options:

- Frames (default). Time values are in units of frames. The number of frames needed to comprise one second depends on the video system setting.
- Real Time. Time values are in units of minutes, seconds, and frames. This is the standard time code system used by video equipment, and is also dependent on the video system setting.
- Seconds. Time values are shown in units of seconds. This system may be more natural if you are doing physical simulations, or are using physical formulas within an animation.

You can change the time system at any time. The user interface automatically adapts to the selected time system. For example, if you select the time code value, three fields (minutes, seconds, frames) are used to show animation timing. Also, changing the time system does not affect any existing motion definitions. The time system only affects how the user interface displays time attributes.

The video system can be selected from the Animation Settings window. You can find this window below the select window in the default startup project, and in the Windows pull down menu.



The time line showing real time units

Suspending and resuming animation rendering

Realsoft 3D allows you to start and stop the rendering of a project or scene at any time, without losing the work that has been performed. This is quite useful when rendering long animation sequences.

Assume that you have started rendering a long animation, say 1200 frames, and you want to stop the rendering for some reason, and then resume it again later from the point at which it was stopped.

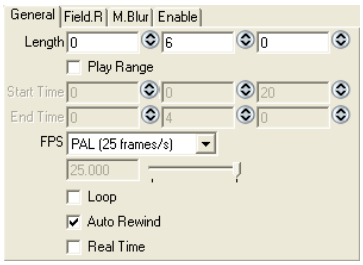
The controls for doing this can be accessed either from the file rendering window, or from the main menu.

To stop rendering an animation, press the Stop button on the file rendering window, or make the following selection from the main menu: Animation/Playback/Stop. Then use File/Save As to save the entire project. The current time is saved with the project. You can then load the project again at any time, and click the Render Animation button on the file rendering window to continue rendering the project.

Defining project-specific animation settings

Every time you create a new animation, you have to define a number of project-specific animation options. For example, you have to decide whether to use motion blur. You might also want to use field rendering. These options can be defined through the Animation Settings window. You can find this window below the select window in the default startup project.

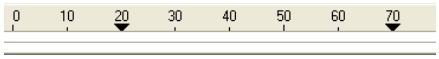
If you select the Motion Blur option, you also need to specify the desired sample count in the Samples field. The higher the value, the better the quality of the generated motion blur (Note that higher values will affect your rendering time).



Project-specific animation settings

If field rendering or motion blur is activated, the animation playback window will show gadgets that allow you to control and select time in units of fields and motion blur samples, as well as frames. This allows you, for example, to access the time line at finer resolutions, such as Frame=2, Field=odd, and Sample=3, and then, for instance, modify an object to create a key frame.

You can also define Start frame and End frame attributes which allow you to play desired sub range of the animation. Check Play range and enter desired start frame and end frame values to activate sub range feature.



Play Range option activated, start and end frame knots shown

This also adds range knots to the time slider. You can drag the knots to adjust the frame range.X

Choreography Basics

In the Animation/Getting Started tutorial, you were introduced to a few of the more common features of the choreography window, such as how to move keyframes, and scale an entire timeline. You were also shown that keyframes you define, whether through the view window or by modifying an attribute on the properties window, end up as a choreograph attached to the object. So far, we have only scratched the surface of what can be done using the choreography tools.

Although it is true that you can keyframe animations to your heart's content using the animation record mode and setting keyframes interactively (through a view or property window), this approach will take you only so far. The choreography system provides a number of tools and methods for creating any type of animation, from simple to very complex (for instance, a combination of simple mouth movements to create *individualized* facial expressions).

This tutorial introduces you to more of the choreography system's features, such as editing aNimation curves, loading and saving choreographs, and building hierarchical animations. More choreography features are presented in the Choreography Essentials tutorial.

To begin, delete all previous test objects from your current scene, and do the following:

Set up an animation timeline of 120 frames.

Set the view window to a front view.

Create a sphere near the left edge of the view window, roughly centered vertically.

Turn on animation record mode and set the time slider to the last frame (the index of the last frame is 119 or 120 depending on the indexing convention selected from the Animation tab of the Preferences window). Then move the sphere in a straight line across to the right edge of the view window. Use the Shift modifier or the red translation handle for accurate horizontal translation.

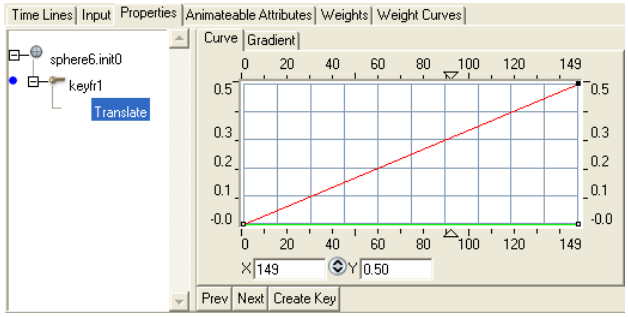
Exit the animation record mode.

Open the choreography window, and click on the Properties tab. Double click the mesh1.init0 choreography. Double click the keyfr1 object under the init choreography. Select Translate from the list of animated object attributes.

Whenever you modify an object in the animation recording mode, the animation system saves all your modifications to a set of animation curves, grouped under a choreograph name. These curves determine how the attribute changes during the animation. The choreography window provides a way to edit these animation curves directly.

Let's have a look at the straight line movement animation we just created for the sphere. In the Properties tab, you should see the following:

For object translation, the red line represents the X axis, green the Y axis, and blue the Z axis. Currently, you see an ascending red line because you moved the sphere in the X direction. The green and blue lines have a constant value zero (the green line hides the blue line below it). Study the graph of the animation curves for a moment. The X axis of the graph shows the number of frames of your timeline, and the Y axis shows object position in the current units. Note the positions of the "knots" of the lines (there should be one at each end on both lines).



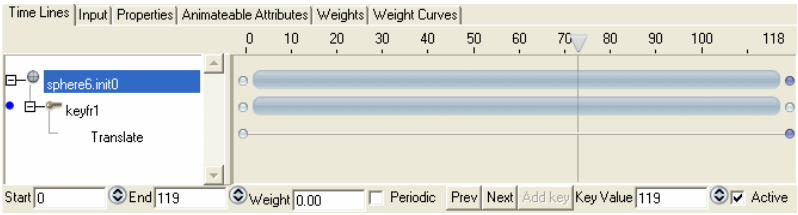
The curves defining the translation of the sphere



Note

The curve display graph can be zoomed and panned by dragging with the right mouse button while holding down the Shift or Ctrl key, respectively. From the popup menu of the curve graph, you can hide any of the red, green and blue curves in order to make editing of a single curve easier.

Now compare what you see in the animation curves graph to the information shown on the Time Lines tab.



A time line showing keyframes for the sphere

Now return to the Properties tab. Try moving the knots on both the X and Y animation curves, and observe the results. As suggestions, do the following:

1. Move the end knot of the red curve to a value of 80 (this changes the value shown in the X field below the graph). Keep the value of the Y field the same. Move the time slider manually to view the animation. Also look at what has changed on the Time Lines tab.

2. Return the end knot of the red curve to its original location, and move the beginning knot to an X value of 40. Run the animation, and then check the Time Lines tab.

3. Move the beginning knot of the red curve to a Y value of 0.3. Note that you can enter values directly in the X and Y fields, and press the Enter key to set the new value. Run the animation, and then check the Time Lines tab.

4. Set the beginning knot of the red curve back to its original position. Now move the end knot of the green curve to a Y value of 0.4, and observe the animation and the changes in the Time Lines tab.

All the manipulations we have done so far are rather basic; however, maybe you can now envision the type of control you have available for creating animations. Moving on to more interesting things, let's assume you want another keyframe for the movement of the sphere along the X axis. You could go back to the view window and use the animation record mode, but this is not necessary. Try the following:

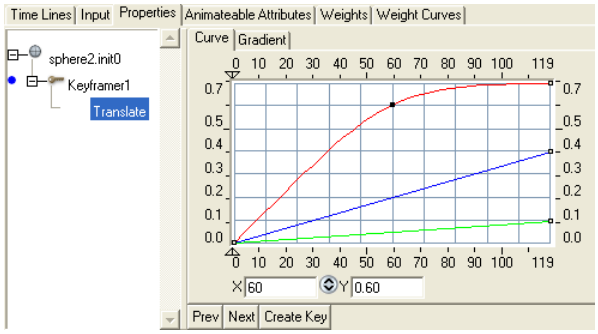
Make sure the knots of both the red and green curves are set back to their original positions. Now place the cursor over the red X curve at the point where it intersects the line representing frame 60, and left click once. A new knot is inserted. Best of all, this new knot is a keyframe. Verify this by viewing the Time Lines tab. Now you can move this new knot, changing the shape of the X curve, and thus, the movement of the sphere along the X axis. Move the new knot to a value of 0.6 for the Y field, and run the animation.

What has happened? The sphere still moves in a straight line, but you have modified how it does so. Now the sphere moves most of the distance across the screen by frame 60, and then takes another 60 frames to travel the remaining distance. The result is as if the sphere has encountered some sort of force field or other type of external resistance. (This isn't really the case, but by adding other elements to the scene you could certainly make your viewers *believe* the sphere is meeting some type of resistance, and isn't that the point?)

Next, let's modify the movement of the sphere along all three axes. Adjust the translation curves to look like the following:

These are the settings:

- X (red) curve is left as it was from the last adjustment. Knot 1: 'X=0, Y=0.0'; Knot 2: 'X=60, Y=0.6'; Knot 3: 'X=119, Y=0.7'
- Y (green) curve settings. Knot 1: 'X=0, Y=0.0'; Knot 2: 'X=119, Y=0.1'
- Z (blue) curve settings. Knot 1: 'X=0, Y=0.0'; Knot 2: 'X=119, Y=0.4'



Modifications to all three translation curves

Now play the animation. First view from a front view. Then rerun it and view it from a top view. Now the sphere moves along all three axes. As you should now start to realize, this form of editing, when applied to other parameters such as Rotation, Scale, Skew, Color, Transparency, etc., gives you very precise control over your animations.

Let's do one more adjustment before closing this discussion, which is, let's make the sphere move along a circular path in the XY plane, using the starting and ending positions that we have defining along the X axis.

To do this, set the end point of the blue curve back to a Y value of 0.0, and set the X curve back to its original state. To do this, select the knot at frame 60 that you added to the red curve (left click the knot to select it). Right click and select Delete Points from the popup menu. Now we have no Z motion, and the X motion is set to what it was originally.

Now we need to do two things: make the X movement return to its starting point at frame 119, and adjust the movement of the sphere in Y so that the sphere goes to the bottom and the top of the screen when the sphere is at the middle of the screen in its movement along X.

To modify the sphere's movement in X:

1. Add a knot to the red curve at frame 60. Move this knot to a Y value of 0.7

2. Move the end knot of the red curve at frame 119 to a Y value of 0.0

3. Right click over the curve graph and select 'Types/Polyline' from the popup menu. If you now play the animation, the sphere moves back and forth across the screen.

To modify the sphere's movement in Y (the green curve):

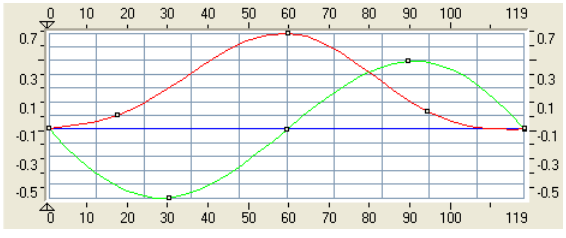
5. Move the end knot of the green curve to a value of 0.2 so you can more easily see the curve. Add new knots to the curve at frames 30, 60, and 90.

6. Set the Y value of the knot at frame 30 to -0.5.

7. Now set the knot at frame 60 to 0.0, the knot at frame 90 to 0.5, and the knot at frame 119 to 0.0

8. Play the animation.

Note that the sphere does not really move along a smooth circular path. This is because we earlier set the animation curve type to Polyline. Right click over the curve graph again and select Type/Nurbs from the popup. Add two new points to the red curve and bend it to a bell shape as shown in the figure below. Now play the animation. The sphere moves along a (roughly) circular path, returning to its starting position.



The final translation curves



Note

The animation system uses special PNurbs curves by default. These curves attempt to pass through the key points as smoothly as possible. If you need to combine smooth, round bends and sharp edges in a same curve, use the Bezier option available in the curve editor's popup menu.



Note

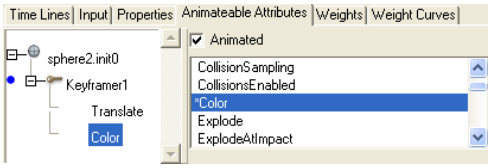
Animated transformations are defined in object's space. The red X curve does not show you movement in 'world' space but with respect to its own native orientation. The red axis of the object handles shows you this direction. Therefore, you can edit animation components of an object independently of the motions of its parent levels. For example, a butterfly moves its wings up and down. When you insert the butterfly to a moving space ship, up direction may become unclear - however, editing the wing motion remains quite as easy.

This section has only scratched the surface of what you can do by editing animation curves. As always, the best course of action is to experiment. Save the project you just completed because we'll use it in the next section.

Adding animated attributes to a choreograph

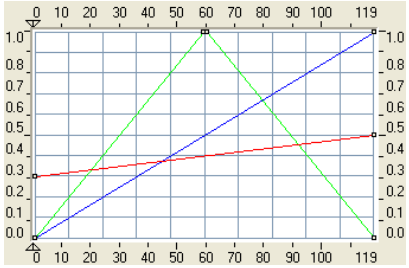
In the previous section, we used the Properties tab to modify an animation that was initially defined interactively. If we now want to add rotation or scaling effects, for instance, is it necessary to do this first by using the animation recording mode? The answer is No. In fact, you do not even have to set up an initial animation in animation recording mode at all (more about this in the next section).

Using the animation we created in the previous section, let's animate the color attribute of the sphere so it changes color as it moves. First, go to the Time Lines tab and select the choreograph 'keyfr1' - we will define the color animation using an existing keyframer object. Then switch to the Animateable Attributes tab, Scroll down the list of attributes, highlight Color, and then click the Animated checkbox. An asterisk to the left of an attribute name indicates that it is animated (even if the animation settings do nothing).



Select the Color attribute from the list and click the Animated checkbox

Go to the Properties tab and click the new Color attribute in the choreography list to select it. Right click over the graph window and select Create Predefined Curve/Constant curve from the popup. Three animation curves are created for the Color attribute: red, green, and blue lines. These lines will be set one on top another initially. At this point, the vertical scale of the curve editor may be too small (0 - 0.5). Select Set Minimum and Maximum Values from the popup menu, change Max Y to 1.0 and click OK.



Curves defining object color

Move the knots of the red, green, and blue curves so that you have something similar to that shown in the following illustration.

Note that for the green color component, two knots were added around frame 60 (the curve type is the default of PNurbs) to achieve straight line segments. However, since all the other color curves are straight lines, you could achieve the same effect by adding only one knot to the green curve at frame 60, and changing the line type to Polyline.

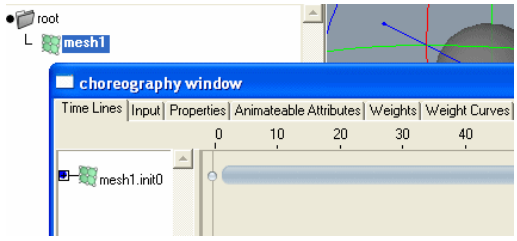
Play the animation. You now have a sphere that changes color as it moves, and you were able to add this effect solely in the choreography editor.

Creating a choreograph for an object

As mentioned in the previous section, it is not necessary to use the animation recording mode to create an initial choreograph for an object. Realsoft 3D's choreography editor can be used exclusively to construct your animations.

To demonstrate this, first delete the sphere from your current scene to get a clean start. We will use the same frame count as in the previous example, 120 frames.

Create a NURBS sphere at the left side of the view window. Select the sphere in the select window, and then right click to display the popup menu. Near the bottom of this menu is the Make Choreographable option. Select this menu, then display the choreography window, and go to the Time Lines tab. There is now an entry called 'mesh*n*.init0'. If you double-click this choreograph, no sub items are displayed because it currently does not have any animated properties.



A new choreograph

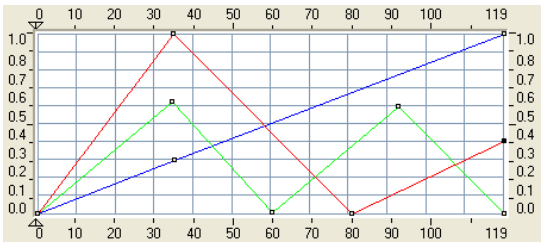
Let's make this sphere expand and contract as it moves back and forth across the view window. First we define the motion to make the sphere move across the view and back to its starting position:

1. On the Time Lines tab, highlight the mesh*n*.init choreograph. Move the end of the time line to frame 119.
2. Select New/Keyframer from the popup menu. Double click mesh*n*.init to open its sub structure.
3. Click the keyfr1 choreograph to select it.
4. On the Animateable Attributes tab, select Translate from the list of attributes, and set the Animated checkbox.
5. Open the keyfr1 choreograph with a double click and select the new Translate attribute under it.
6. On the Properties tab, select Create Predefined Curve/Constant curve from the popup of the curve graph.
7. Select Type/Polyline from the popup.

Let's change the editing area of the curve graph. Right click over the graph and select Set Minimum and Maximum Values. Set the fields as follows: Min X = 0; Max X = 119; Min Y = -0.8; Max Y = 0.8. Click the OK button. Uncheck Green and Blue curves from the popup menu; only the red curve remains visible. Click on this line at frame 75 to add a new knot, and move it upwards to a Y value of 0.8 (you may need to adjust the Y value to match the scale of your view window. The main point is to have the sphere move from one edge of the view to the other and then back).

Now let's animate the sphere's scale. Find the Scale attribute in the Animateable Attributes tab, select it and check Animated. Return to the Properties tab and select Create Predefined Curve/45 deg curve from the popup. Select Type/Polyline from the popup. Add points to the X and Y scale curves so that they look as follows:

Now play with the time slider to see how the scale curves affect the shape of the object. The Scale curves, as well as Translation curves, define a change from the initial state. This means that when a scale curve reaches the Y value 1, the total scale value is 1+1=2 (the initial default scale is one), doubling the size of the object in the respective dimension.



Animate the Scale property

Loading and saving choreographs

So, you have created a nice animation that moves an object and causes it to "pulse", increasing and decreasing in size. What if you wanted other objects to do the same thing, or wanted to save this motion into a library of animation effects? Well, choreographs can be saved to a file. This allows you to create choreography libraries.

To save animation data of an object to a file, such as scaling and motion of the sphere created earlier, select the top level chorograph of the sphere (mesh.init0) and choose Save from the pop-up menu. Save it as 'Pulse'.

Now create another object, for example an analytic cube. On the select window, choose the popup menu Make Choreographable. On the choreography window, select the init choreography of the cube. Then select Load from the popup menu. From the file loading dialog, select the 'Pulse' choreograph you just saved, then play the animation. The cube moves and pulses in exactly the same way as the sphere did.

Rules for Attaching Choreographs to Objects
In general, any choreograph can be attached to any type of object even if the new target object does not have a common set or subset of object attributes. When a choreograph is attached to a new target object, all features that the target object does not support are removed. For example, an analytic sphere object defines a Center attribute, but a NURBS curve does not have a Center attribute. A choreograph created for an analytic sphere that only contains animation of the Center attribute would produce no effect if assigned to a NURBS curve.

Deleting animation effects

If you are working with animating an object, and decide that a particular choreograph is not what you want, you can easily delete it from the object. If a choreograph has been saved as a file previously, deleting it from an object does not delete the choreograph file, just its attachment to a particular object.

Using our example from the previous sections, if you want to delete the 'Pulse' choreograph from the cube object, do the following:

1. Go, for example, to the Time Lines tab of the choreography window and select the keyfr1 under the mesh.init0 choreograph.

2. Select Delete from the pop-up menu.

You can also remove animation effects from a certain attribute. For example, maybe you want the Pulse choreograph just for the scaling that causes the pulsing behavior, and not the movement. To remove the defined animation effects from the Translate attribute:

1. In the choreography window, click the Translate attribute under the keyfr1 choreography to select it.

2. Select Delete from the popup menu.

You can also remove all choreographs from an object. To do this, select the topmost 'init' choreograph and apply Delete from the popup menu. Alternatively, you can activate the object from the select window and choose Delete Choreographs from the popup menu of the select window.

Exchanging choreographs

In the tutorials above, a choreography was assigned from a NURBS mesh to a cube by saving and reloading it. You can also assign a choreography to a new object using drag & drop.

For example:

1. Create a cube. Turn animation recording on, go to the last frame, then rotate and move the cube. Turn animation recording off.

2. Create a sphere. Select Make Choreographable from the popup menu of the select window.

3. Multi select the sphere and the cube or their parent level (because chorerography window may have the Only Selected Objects option enabled).

4. Open the choreography window. Open the Cube.init0 choreography by a double click. Drag the keyfr1 choreography under it over the sphere.init0 choreograph and drop it there. This attaches the dragged choreography to the sphere.

If you now play the animation, both objects will follow the same choreography.

Choreography Essentials

Each animation choreography can be controlled through a set of parameters. Therefore, in many cases it is often better to record different kinds of animation effects as separate choreographs. For example, a bouncing sphere animation can be accomplished by combining two motions: the *forward* movement and the *bounce* movement. If created in this way, such choreographs can be used to make many types of "bouncing sphere" animations by editing the parameters of the choreography rather than keyframing a new animation.

In addition to its time line, a choreography has a number of other parameters that can be used to control timing.

Parameter Name	Description
Weight	Controls how strongly a certain choreography affects an object. For example, you could create a mouth movement that consists of 10% of the 'smile' and 40% of the 'open mouth' choreographs.
Periodic	Repeats the choreography. The time line is used to define the start and end times for one period (cycle). For example, record one 'bounce' cycle for a sphere, and make it bounce all the time by setting this option.
Active	Turns a choreography on/off. Often, it is easier to add new motion components when other components are turned off.

The weight parameter can be animated. For example, a talking head can be created by animating the weight parameters of various basic expressions.

Animating weights

Each choreography can be assigned a *weight*, which controls how strongly it affects an object. To see how this works in practice, we'll create a sphere with motion defined by two separate choreographs.

1. Create an analytic sphere near the top left corner of the view window.

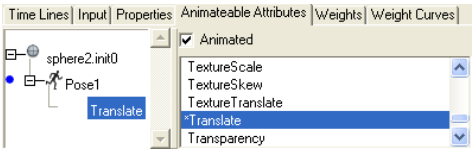
2. Select Make Choreographable from the popup menu of the select window.

3. Open the choreography window and select the 'sphere.init0' choreography. Select New/Pose from the popup menu of the choreography window.

4. Double click the init choreography to open its sub structure. Select the pose choreography under it.

5. Go to the Animateable Attributes tab, find and select Translate from the attribute list and check Animated.

6. Move the sphere horizontally near the top right corner of the view window. You have now created the first choreography for the sphere. Apply the Rename tool of the popup menu to change the name of the pose choreograph to 'right'.



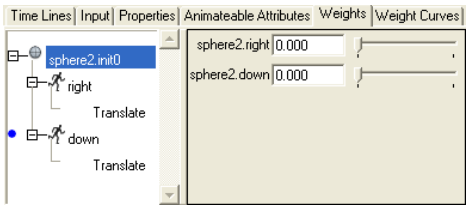
Add Translate attribute to the pose

7. Go to the Weights tab and try the sphere.pose1 weight slider. The sphere moves between its original position and the new position. Leave the weight slider to zero.

8. Select New/Pose again. By default, it inherits the attributes of previous choreographs. The new choreography is automatically the current choreography (a little yellow triangle on the choreography window marks it), and therefore new posing actions will be stored to it. So, just move the sphere vertically to the bottom left corner of the view window. Rename the second choreography as 'Down'.

We now have two choreographs. If you now play with the time slider, nothing happens, because the poses just define static attribute states, which do not depend on the time.

By combining these two choreographs with different weights, you can create 'bouncing sphere' animations easily. For example:



Two choreographs created for the sphere

9. Select New/keyframer from the popup menu. Then enable animation recording. The weight modifications presented below will be recorded to the keyframe choreography.

10. In the first frame, set:

- sphere.right = 0
- sphere.down= 0

Since the effect of both choreographs is now zero, the sphere returns back to its original position near the left top corner of the view window.

10. Move the time slider to half way in time and set weight sliders to:

- sphere.right = 0.5
- sphere.down= 1

This takes 50% of 'right' position and 100% of 'down' position.

11. Move the time slider to the last frame. Then define:

- sphere.right = 1
- sphere.down= 0

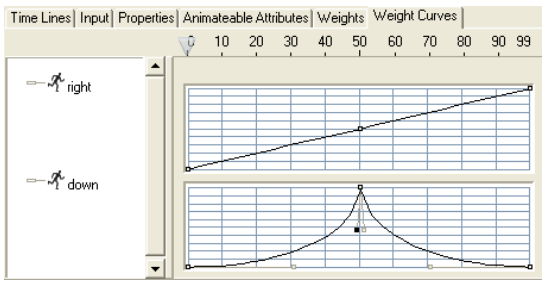
12. Turn animation recording off and play the animation. You can now see the result of the two blended choreographs: the sphere travels along a curved path across the screen.

We can improve the result by editing the weight curves as follows:

13. Go to the Weight Curves tab of the choreography window. You can now see two curves. They indicate how strongly each choreography affects the objects as the time changes.

14. You can edit these curves directly to fine tune the blending of the choreographs. The weight curve for 'right' choreography is OK, but change the type of the 'Down' curve to Bezier from the popup menu of the curve graph. The figure below shows a better weight curve for the 'down' choreography - it is easy to edit the Bezier curve to this kind of shape.

The strength of the first choreography increases with time. In other words, the 'right' choreograph affects more and more as the time proceeds, making the object to move vertically. The 'down' weight curve has a peak in the middle of the animation. This makes the sphere to fall down and bounce back. The combination is a moving and bouncing sphere.



Bouncing created by animating weights of 'right' and 'down' choreographs

Hierarchical animations

In the modeling specific tutorials we have demonstrated how geometric objects can be constructed to hierarchical structures using level objects. For example, you can create a 'leg' object, which consists of two sub objects: a thighbone and a shinbone.

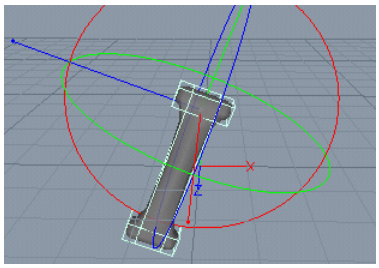
Correspondingly, you can attach a choreography to the entire level object and another choreography to the its children.

To demonstrate this, let's create walking legs.

Tutorial level: Medium

Example project: 'tutorprojects\animations\gettingstarted\walking legs'

First, we need to create a leg consisting of two sub objects: a shinbone and a thighbone.



A simple shinbone in the initial 'forward' position

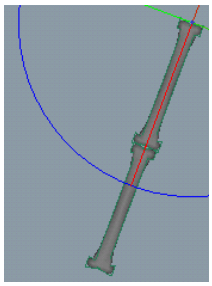
1. Create a simple object representing shinbone (for example, edit a SDS cube). Move the pivot point to the kneecap (the leg object will be rotated about the knee).Tip: you can use the Insert key to activate the pivot move tool.

2. Luckily the thighbone looks pretty much like the shinbone. So, duplicate the shinbone and move it upwards so that the lower endpoint of the thighbone matches the upper endpoint of the shinbone.



The structure of the leg

3. Now select both the objects and apply 'Drop to a level' from the select window's pop-up menu. This creates a new level object and drops the selected bones inside the level. You can rename the new level object as 'left leg'.



4. Move the pivot point of the 'left leg' to the hip. We do this because the entire leg will be rotated about the hip.

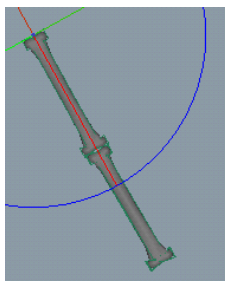
The hierarchical leg model, in its initial position, is now ready. Let's animate it.

5. Set animation recording on, jump to frame 20 and use rotation handles to rotate 'left leg' about its hip so that the leg now represents its back position.

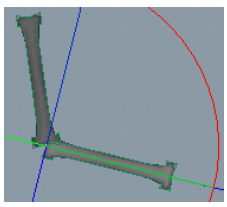
6. Jump to frame 40 and rotate 'left leg' back to its original position.

7. Now, jump to frame 30 and rotate the shinbone so that it represents the back position.

8. Go to frame 40 and rotate the shinbone back to its original position.



The whole leg rotated backwards in frame 20

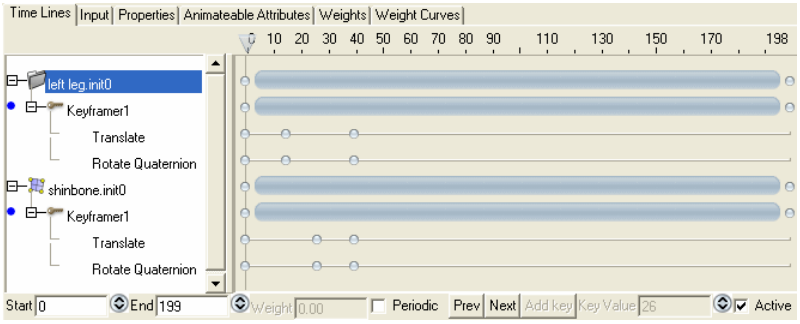


The shinbone rotated in frame 30

Reset animation recording and play the animation. The shinbone rotates about the kneecap while the entire leg rotates about the hip. Because of hierarchical construction, the parameters used for rotating the shinbone are rotated with the thighbone.

However, the leg takes only one step! As soon as the time reaches the last key frame 40, the leg stops. How can we make the leg walking all the time?

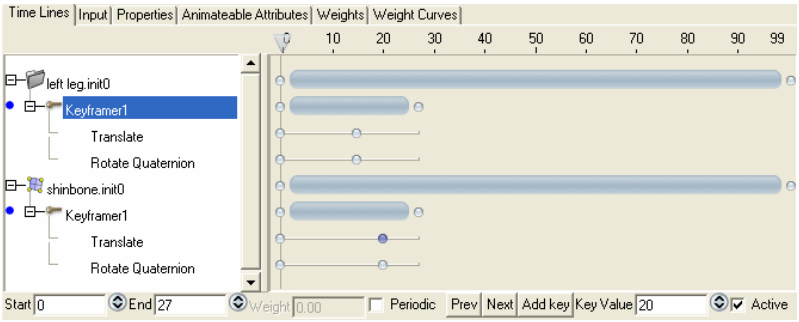
9. Select the left leg object and open the choreography window. Go to the Time Lines Tab. Open the time lines of the left leg and the shinbone, as well as the time lines of the included keyframe choreographs. Although the time lines cover the entire animation, there are keys only during the first 41 frames.



Choreographs rotating the left leg and shinbone

10. First we normalize the time lines so that they correspond the actual key range. Click the end knob of keyfr1 timeline under the left leg:init0 timeline. Press Shift key down and click the end knob of the keyfr1 timeline of the shinbone object. Now both end knobs should be selected. Press also the Alt key down; then Alt-Shift-drag the end knobs to the frame 40. Then you can release the hotkeys. Dragging with the Alt key down scales a timeline but does not scale the underlying keys.

Note that it does not matter, which choreography item is selected when you drag the time line knobs in the time line graph. However, the numeric Start and End fields at the bottom of the window show the properties of the selected choreography, and therefore selecting a choreography before dragging its time line knob helps to make the operation accurate.



Keyframe time lines normalized and made periodic

11. Select the left leg/keyfr1 choreography and set the Periodic option from the bottom of the choreography window. Set Periodic for the shinbone/keyfr1, too. Play the animation. The leg object now repeats its motion several times.

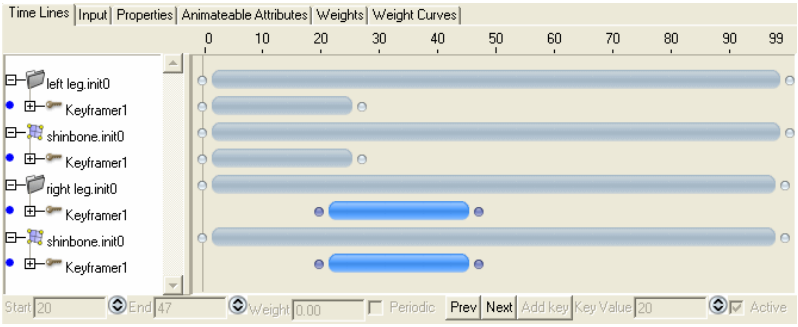
How to create another leg?

12. On the select window, select the left leg object and apply the Duplicate tool from the popup menu. Rename the copy as to 'right leg'.

Now we have a new problem. We have two identical legs, which also function identically.

13. Multi select the left leg and the right leg. Go back to the choreography window. Select the choreography right leg/keyfr1. Then click in the middle if its time line to select it. Shift click the time line of the choreography right leg/shinbone/keyfr1 to multi select it, too. Then Shift drag both time lines to the right until they start from the frame 20.

14. You can now close the choreography window. As the final step, take a top view and move the legs slightly apart from each other. Play the animation to see how the legs walk.



Time lines of the right leg shifted

Path Animations

In the Getting Started tutorial, you animated a sphere that moved roughly in a circle - this demonstrated how to create a choreography based on *keying* an object's position at various times. As you may have noticed, to refine the movement to be circular, you had to make adjustments to the animation curves in the choreography editor. Although this method is fine for certain types of movement, Realsoft 3D provides other options. One of these options is to animate an object along a path, hence the term, *path animation*.

This tutorial explores some of the ways you can create path animations, which includes attaching an object to a two dimensional object such as a line or a three dimensional object such as a surface.

Tutorial level: Medium

Example project: 'tutorprojects\animation\paths\moving sphere'

Remember the animation of a sphere traveling in a circle you created in the Getting Started with Animation tutorial? Now let's do that same animation using a path.

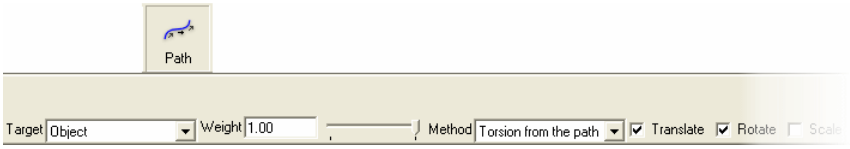
First, set up your environment. Then create an analytic sphere in the upper left corner of the view, followed by a NURBS circle centered approximately at the view's midpoint with a diameter of about 3/4 of the view's size.

Select the sphere and then the circular curve (the order is important), and then choose Tools/Lattice Mapping/ Create Path Animation from the main menu (or, if you are using the startup environment, open the Lattice Mapping tab, and click the Path button. This presents a control bar with the path tool's options.



Note

It is important that you select the objects in the specified order. If you select the objects in reversed order, you would get a curve moving along the surface of the sphere.



Options for the Path tool

For this first attempt, use the default settings, so just click Accept. Now play the animation and note what is occurring. First, the sphere follows a circular path based on its initial position and its offset from the path curve. Second, the sphere rotates about the center point of the circle as if there were an invisible rod connecting the center of the circle to the center of the sphere. To contrast this rotation, double click the sphere object to open the property window, go to the Map tab and uncheck the Rotation attribute, then replay the animation.

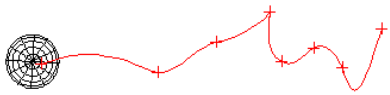
Object on an open path

Tutorial level: Medium

Tutorial project: 'tutorprojects\animation\path\morphing path'

Making an object follow a path defined by an open curve is essentially the same as using a closed curve. Once again, the initial positions of the two objects, in relation to each other, determines the actual movement (that is, if you are using the Translate option).

Set up a sphere and a curve as shown in the image, and use Lattice Mapping/Path with the default options, and then play the animation.



A sphere and an open NURBS curve as the path

Now edit the NURBS curve, maybe straightening out the dip near the middle, and then play the animation. Try just moving points first, then edit the curve by also removing points. Next, add some new points to the curve, and reshape it again. Note that when you make these changes, the sphere speeds up and slows down as it traverses the path. This is based on the spacing of the curve points: the greater the distance between points, the faster the object moves along that line segment. No matter what type of editing you perform on the line, the object remains correctly attached to the path.

Now try editing the path curve while in animation recording mode. Select the curve and enter Edit mode. Set the time slider to frame 0 and activate animation recording. Move to frame 30 and point edit the curve. Move to frame 60 and adjust the curve some more. Continue editing the curve at frame 90, then deactivate animation recording. You now have a sphere that moves along a morphing path.

Unattaching an object from a path

As you may guess at this point there are quite a number of ways, based on the relative position between the object and the path, that you can set up a path animation. So what if you attach an object to a path, and you do not like the result? You can easily *unattach* an object from a path with the Unmap tool.

Select the object and then the path. Then, from the Lattice Mapping tab, click the Unmap button. The object and the path are no longer attached. You can now modify the positions of either object and connect the two again.



Note

You can also access the Unmap function from the main menu, Tools/Lattice Mapping/Unmap Objects.

A swimming fish

This example demonstrates how to create swimming fishes, wriggling snakes and other effects where the motion path should not only transform the object to be animated but also deform it.

1. Create a fish by using NURBS or subdivision tools. The only requirement for the fish is that it contains a sufficient number of points so that the curve can deform it.



2. Create a NURBS curve representing the path.

3. Select the 'fish' and the 'path' (in this order). Then click the Path tool. This time set the Target option to Points and click Accept.



Select Target=Points



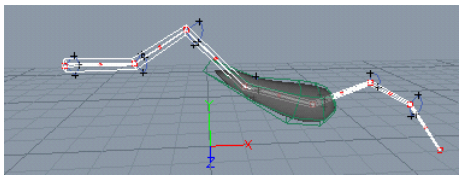
A freeform fish and a NURBS curve

Play the animation and the fish wriggles along the path. Keep this project open for use in the next section.

Reconnecting an object to a different path

Tutorial level: Medium

Tutorial project: 'tutorprojects\animation\path\swimmingfish'



The fish moving along a skeleton

The Lattice Mapping/Reconnect tool allows you to exchange one path for another.

Using the swimming fish project from the previous section, add a skeleton object to use as the path for the swimming fish. Select the fish and then the skeleton, and then activate the Reconnect tool. Click Accept and the fish jumps to the skeleton. Move the time slider and the fish moves along the skeleton.

Example project: 'tutorprojects/animation/paths/fish on a skeleton'

Mapping specific points to a path

Tutorial level: Medium

Tutorial project: 'tutorprojects\animation\path\ik_path'

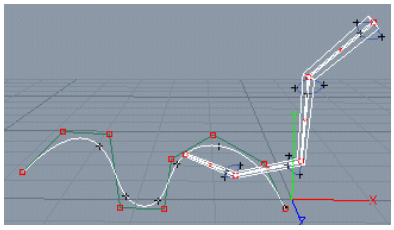
So far we have dealt with mapping entire objects to paths, but what if you only want a part of an object to move along a path, for instance, the end joint of an arm. For this you set the Path tool to use the Selected Points value for the Target option.

For example, you can use a NURBS curve to control the path of a joint of a skeleton:

1. Create a skeleton and a NURBS curve defining a path for a joint of the skeleton.

2. Multi select the skeleton and the path (in this order) and enter the Edit mode. Now, select the end point of the skeleton by clicking it.

3. Activate the Path tool and set Target to Selected Points. Click Accept.



The last joint of a skeleton attached to a NURBS curve using the path tool and the Selected Points option

You have now attached the joint to the path. Play with the time slider and see how the joint follows the path. Note that the skeleton uses inverse kinematics to follow the path, which is what you might have already expected.

Timing

In the previous tutorials we have used two key frames to set up path animations. In the first frame the object is at the start of the curve and in the last frame the object is located at the tail of the curve. The intermediate positions are interpolated for us by the animation system.

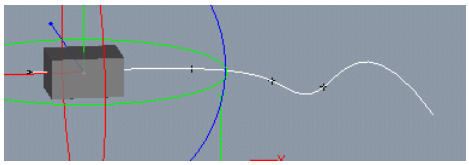
The speed at which the object travels between the two ends of the path depends on the properties of the path. Depending on the point density and parametrization of the path, the object may accelerate and decelerate while it moves along the path.

You can control the default speed and timing very easily by adding new key frames. This tutorial explains how to do that.

Tutorial project: 'tutorprojects\animation\path\moving and rotating cube'

1. Set the frame count to 200. Create an analytic cube and a NURBS curve to define a motion for it.

2. Select the cube and then the NURBS path. Apply the pull down menu Animation/Path/Move. Play the animation to verify that the cube travels to the end of the path during the playback.



A cube follows a path

Let's change the motion so that the cube reaches the end point of the path at the frame 100 and then travels back to the start of the path.

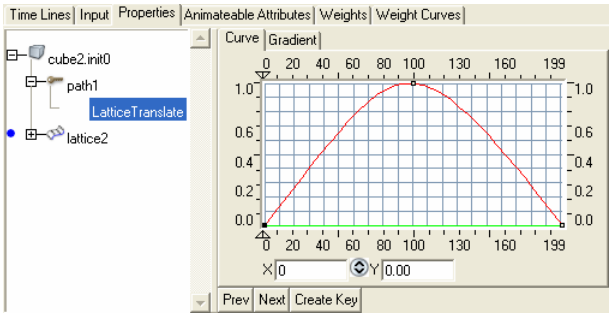
3. Open the choreography window (pull down menu Animation/Choreographs..). Go to the Properties tab. Open the sub structure of the 'cube:init0' choreography. Open also the structure of the path choreography. Select the attribute LatticeTranslate. The curve graph at the right side of the choreography window now shows a linearly ascending red curve. The green and blue curves have a constant zero value.

4. Click in the middle of the red curve and move the inserted point up to level y=1.0.

5. Move the end point of the red curve down to zero level.

Play the animation. The cube travels to the end of the path, slowly decelerates and then travels back. Deceleration at the end point is a result of the round curve shape. If you change the curve type to polyline, direction changes without deceleration.

Note that the red curve defines the position along the path. You can also create some key frames to the green and blue curves to temporarily displace the cube from the path.



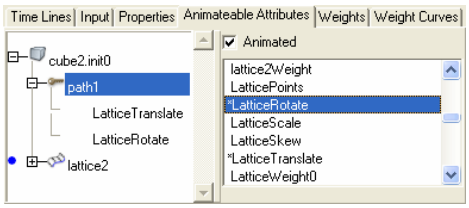
A lattice translation curve, which makes the cube to return to the start point of its path

In addition to translation, you can naturally control how the cube rotates in the path object's space. For example, to make the cube rotate while it moves on the curve:

6. Go to the Animateable Attributes tab on the choreography window. Select the path choreography.

7. Find and select LatticeRotate from the attribute list. Check Animated option for it.

8. Go to the Properties tab. Select the new LatticeRotate attribute under the path choreography. The curve graph shows no curve at this point. Select Create predefined curve/45 degree curve from the popup menu of the curve graph.

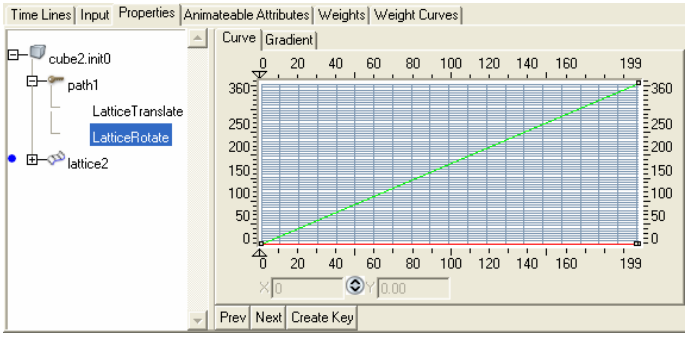


Make Lattice Rotation animated

9. Select Set Minimum and maximum values from the popup menu. Set Max Y = 360 degrees, check rescale and hit OK.

10. Move the end point of red curve and blue curve down to zero level. The green curve will grow from zero to 360 degrees, meaning that the cube will make exactly one full revolution around the path during its travel.

Play the animation. The cube moves and rotates in the parameter space of the curve.



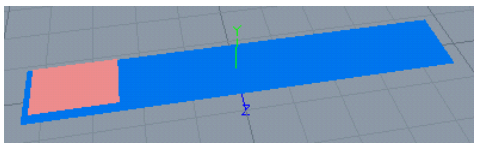
Lattice Rotation curves

Deformation paths

Let's assume we want to create a 'flying carpet' animation. This can be done by using a NURBS mesh as a path.

Tutorial level: Medium

Tutorial project: 'tutorprojects\animation\paths\mesh path'



A red 'carpet' mesh on a blue 'path' mesh.

1. Create a carpet (say a NURBS mesh consisting of 5 x 5 control points).

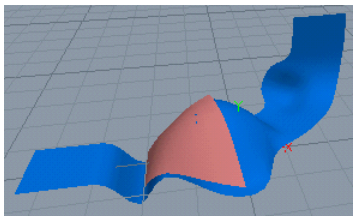
2. Create a bigger NURBS mesh consisting of 20 x 5 control points. This mesh represents the path of the carpet. We want the carpet to move on the surface of this mesh.

3. Select the 'carpet' and 'path' meshes, activate the Path tool, set the Target option to Point and click Accept.

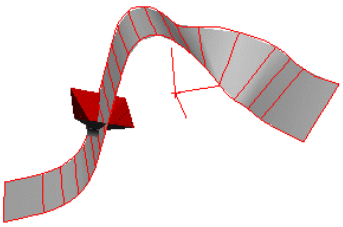
If you now move the time slider, you can see how the carpet moves along the blue mesh.

4. Modify the path mesh so that it contains bumps here and there.

Play the animation to see how the carpet moves along the curved path mesh.



The carpet follows a curved path mesh



Camera following a NURBS stripe



Note

A NURBS mesh as a camera path allows very good banking control.

Tutorial project: 'tutorprojects\animation\path\camera moving along a NURBS stripe'

Mapping objects to curved paths

Tutorial level: Medium

Tutorial project: 'tutorprojects\animation\path\map to curved path'

Let's assume we have a very curved surface which we need to use as a motion path for an object. For example, we need to move a mesh on the surface of a NURBS head.

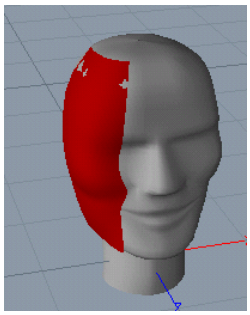
There are two problems. First, we need to deform the mesh to match the surface of the head at a certain area. If the head is very curved, this is not a trivial job. We could use collision detection or some other tools but if we have a very curved object containing tentacles and holes, matching the mesh on the path mesh can be very time consuming job.

The second problem is that lattice mapping to very curved lattice objects is not always reliable and points can get mapped to wrong lattice coordinates.

The Reconnect tool solves this problem. You can first map the object to a planar mesh to define appropriate lattice coordinates. You can even animate the object on that well behaving planar lattice object. Then, use the Reconnect tool to swap the path object to the final curved surface.

Let's see how this works.

1. Create a NURBS mesh consisting of 12x12 control points and an analytic rectangle on which we will animate the mesh. Then map the NURBS mesh to the rectangle exactly the same way as we mapped the NURBS carpet to the path mesh in the previous example.



A red mesh moving on the surface of a head

If you now play the animation, the mesh moves along the analytic rectangle.

2. Activate animation recording mode. Modify the mesh at suitable frames using move, rotate, etc. tools to create some key frames.

3. Load in the actual object you want to be used as a motion path (for example, 'models\nurbs\simple head'). If you use File/Load, remember to uncheck Replace, otherwise you will loose the project!

4. Select the mesh and the NURBS head and activate the Reconnect tool. Click Accept and the mesh jumps on the surface of the NURBS mesh!

Now that the mesh follows the shape of the NURBS head precisely, you can add some further key frames, fine tune motions, etc. without any problems.

Multiple motion paths

Tutorial level: Medium

Example project: 'tutorprojects\animation\path\multiple paths'

The lattice mapping system allows you to map an object to multiple lattice objects. For example, you can control the head of a mesh using one skeleton and its tail using another skeleton. Similarly, you can animate an object using several paths.

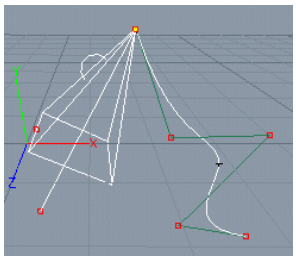
As an example, we will define an animation where the camera position follows a smooth path, but the aim point follows another polygonal path:

1. Create a camera and a smooth NURBS curve representing a motion path for the camera position.

2. This time we don't map the entire camera on the path but its position handle! So, enter the Edit mode and select the position handle of the camera. Multi select the camera and the NURBS path (again in this order).

3. Activate the Path tool and set Target to Selected Points. Then click Accept.

We have now mapped one point of the camera object to a motion path.



Map the camera position handle to a NURBS curve

4. Create a polygonal NURBS curve representing a motion path for the aim point and map the aim point handle to it exactly the same way as you mapped the camera position.

That's all. Play the animation and see how the aim point moves along one path while the position of the camera follows another path.

Animating Materials

To animate one or more materials, you follow exactly the same rules used to animate any other object type in Realsoft 3D. This tutorial presents a number of example projects that demonstrate common material animation cases. We'll start with a simple case, and move on to more interesting ones at the end of this tutorial.

Animating a texture map

Tutorial level: Medium

Example project: 'tutorprojects\animation\materials\moving texture'

One way to animate a texture is to make the entire texture map change in some way. In general, this would do nothing to the characteristics of the texture itself - that is, attributes of the texture - but it would still produce certain useful effects.

To begin, imagine you have a scene that requires a painting on a wall, and you want to create an animation where the painting moves across the wall (maybe this is a haunted house). To do this, you do not need to animate actual material attributes. Realsoft 3D uses geometric *mapping objects* to assign materials to actual geometric objects. This type of system allows you to animate all *geometric* material properties, such as position, scale, and rotation, as easily as animating these same properties of any "regular" geometric object using the animation recording mode or choreographs.

To demonstrate this technique, start by creating an analytic cube as a wall (a front view is probably best). Now do the following:

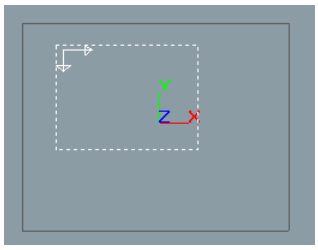
1. Go to the Select window's Materials tab. Right click, and from the popup select New/From Template.

2. The Load dialog opens up. Open the materials folder. Open the Image Maps folder and then select the Color Map file. This creates a new material called 'Color Map'. Find this material in the list, select it, and then rename it to something more suitable, such as Lake_painting, for example.

3. Double click the Lake_painting material to open the property window. Set the value of the File Name field to 'lake,' found in the Realsoft 3D 'textures' directory. Leave all the checkbox attributes unchecked. Close the property wndow.

4. Now highlight the Lake_painting material, then right-click and select Map/Parallel from the popup. Draw a rectangle over the wall object.

5. Ray trace render the view.



Texture map drawn inside the wall object



A material parallel mapped on the wall

6. Set the time slider to the first frame, and activate animation recording. If the "painting" (that is, the texture map object) is not placed where you want it, move it to the starting position. Go to the last frame of the animation, and move the painting to a different location, for instance, the other side of the wall. Turn off animation recording and play the animation. The parallel mapping object representing the position of the painting moves across the wall.

Essentially, that is all there is to it. You can animate a texture map object in many other ways, such as scale, rotation, skew, etc., just like any other geometric object.

You can create many interesting and useful animation effects by animating mapping objects this way. For example, you can animate the Invisible in photorealistic rendering attribute to enable and disable rendering of a certain material. In this way you can create "wink out" effects, or make a transparent portion of an object become opaque. You can even apply gravity and other simulation effects to mapping objects.

Animating texture attributes

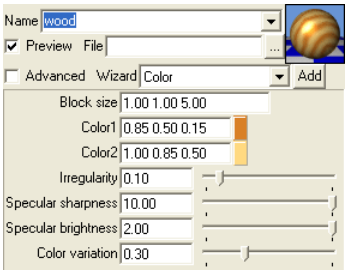
Tutorial level: Medium

Example project: 'tutorprojects\animation\materials\animated wood'

In the previous section we looked at animating a texture map object, but this is not the only way to animate textures. You can also animate any of the attributes of a texture.

For example, let's create a wooden tabletop and animate several characteristics of the wood texture. To do this, you animate the actual material object, not the mapping object.

1. In a top view, create an analytic cube, and size it so it looks like a tabletop. Leave the cube selected.



Wood in the first animation frame

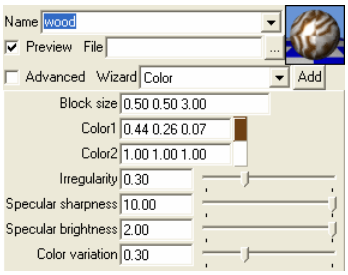
2. Go to the Materials tab of the Select window. Find the standard Wood material and double click its icon to open the property window.

We will use the default values of the wood material for its initial look, so do not change the material in the first frame.

3. Now move the time slider to frame 60 and activate animation recording. Set the Color 1 attribute to a darker brown, and the Irregularity attribute to 0.3.

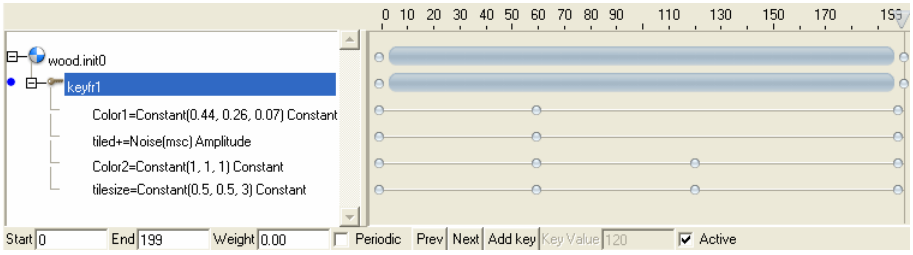
4. Next, move to frame 120 and set the Color 2 attribute to an off-white, and the Block Size attribute to (0.5 0.5 3). Finally, reset animation recording to off.

5. Reset the animation to the first frame, and drag the time slider (do not press play!) through to frame 120 while watching the attribute values change in the property window. Do you notice something? Did you want the Color 2 and Block Size attributes to start changing at the first frame or at frame 60?



Wood in the last animation frame

6. If you wanted the animation of the Color 2 and Block Size attributes to begin at frame 60, this is easy to correct. Open the choreography window (from the pull down menu Windows/Choreography Window) and select the Time Lines tab. Open the wood.init0 entry and the keyfr1 entry under it. Now you see a list of the animated attributes of the choreograph. The last two elements should be Color2= ..... and tilesize=..... Move the first key of these two attributes to frame 60. Now go back, make sure the properties window is open, and drag the time slider again. Watch the attribute values for the animated wood material change over time.


0 10 20 30 40 50 60 70 80 90 110 130 150 170
wood.init0
keyfr1
Color1=Constant(0.44, 0.26, 0.07) Constant
tiled+=Noise(msc) Amplitude
Color2=Constant(1, 1, 1) Constant
tilesize=Constant(0.5, 0.5, 3) Constant
Start 0 End 199 Weight 0.00 Periodic Prev Next Add key Key Value 120 Active

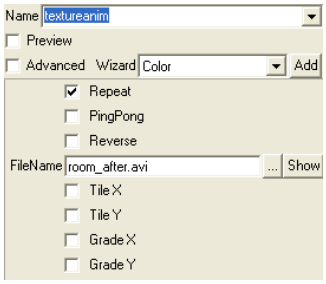
Resetting the keys to frame 60 for the Color2 and tilesize attributes

7. To finish this tutorial, close the choreography window and drag & drop the wood material from the Select window to the view window. This maps the animated wood to the tabletop.

Using animation files as textures

Another method of creating an animated texture is to use an animation file, such as an AVI, as a texture map. This allows you to use, for instance, a video sequence, or a series of images created in another program and rendered as an FLC or AVI file, as a texture.

To do this, go to the Select window's Materials tab, right-click and from the popup select New/From Template. From the Load Selected Objects dialog, open the Animated Image Maps folder, and then select the TextureAnim file. This places a new material called 'textureanim' in the materials list. Rename this material to something useful for your project, and then double-click this material to open the property window.



Property window and attributes for a texture animation material

Enter the name of a file in the File Name field (this should be one of the supported animation file types). Now map this material to an object.

As you can see from the attributes, you have additional options for controlling how the animation file is positioned and played. For instance, you can keep repeating the animation file from beginning to end, you can ping-pong it (beginning to end, end to beginning, and so on), or you can have it play in reverse.

Controlling the timing of an animated texture

The technique we just described works fine if the AVI (or other type) file plays the images exactly at the speed you want. But what then if that is not the case? What if you need an animated image to change first slowly and then speed up? Obviously, you can not achieve this type of control with the basic Ping Pong, Repeat, or Reverse options. The Current Frame attribute is provided for situations like this.

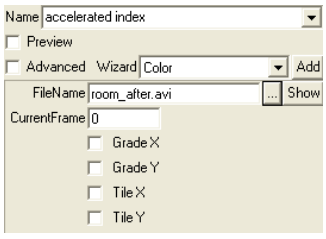
Assume you have an AVI animation file consisting of 10 frames, and that you want to control the speed of the AVI file in special way. For the first 60 frames of the animation, you only want to use the first 3 frames of the animated texture, then during the last 40 frames you want to use the remaining 7 frames of the animated texture.

You set up this kind of an animation by key framing the texture frame index (the Current Frame attribute). To do this, you can use the supplied template material in the Animated Image Maps folder.

1. If you have, or can easily make, a 10 frame AVI, do so and use it in this exercise (see the manual chapter about File Rendering for details).

2. Go to the Select window's Materials tab, and choose New/From Template from the popup. Load the 'Animated Image Maps/Keyframe Texture' material template.

3. The materials list should now contain an entry called 'keyframe texture'. Rename this material template to Accelerated Index, or something else meaningful to you, then double click it to open the property window.



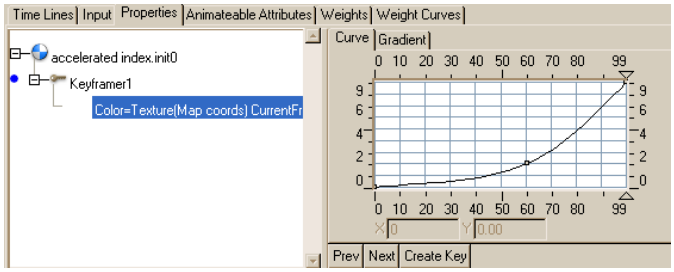
Use an .avi file as a texture

4. In the File Name field, enter the name of your AVI file.

5. Now activate animation recording and set the time slider to the first frame. The Current Frame field is, by default, zero, which is the correct start value (The frame index for AVI files typically begins at 0, so a 10 frame AVI uses index values of 0-9). Move the time slider to frame 60 and set the Current Frame field to 2. Now move to frame 99 and set the Current Frame field to 9. Turn off animation recording.

That's all there is to it: you have now keyframed the *frame index* of your AVI file.

6. Last, open the choreography window to the Properties tab. Select the CurrentFrame attribute from the choreography list to view the animation curve.



The animation curve controlling the frame number of the AVI file as a function of time

Animating Particle Effects

Just like you can attach a material object to a geometric object to define optical and other properties, you can attach 'post effect' objects to particles to render the otherwise primitive particles as more interesting objects.

If you have read the previous animation tutorials, you should already know how to animate geometric properties of any kind of objects, including particles. In this tutorial we will examine how to animate the actual particle effects.

Tutorial level: Beginner

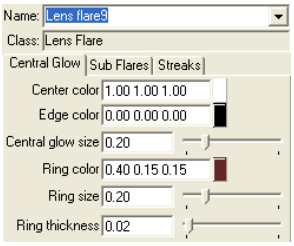
Example project: 'tutorprojects\animation\particles\animated lens flares'

Let us create some 1D particles, which are rendered using an animated lens flare effect. You can create spinning lens flares, star filters, and many other fancy effects this way.

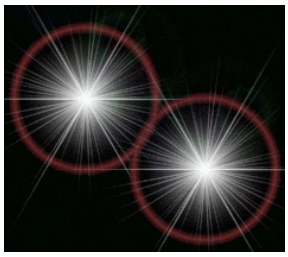
1. Use the particle tool to create some particles. If you haven't used the particle tool yet, the workflow goes as follows: Go to the Particle tab in the toolbar. Activate the particle tool, left click and drag the mouse across the view and select Accept.

2. Select the Lens flare effect from the Select window's Post Particle Effects tab and open the property window.

3. Move the time slider to the last frame. As usual, activate the animation recording mode and modify various lens flare attributes to create key frames for them.



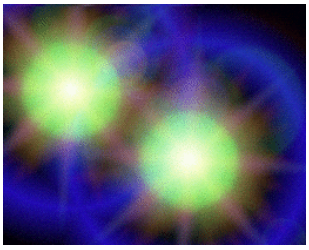
Lens flare properties



The initial state of lens flares

4. Reset animation recording and play with the animation slider to verify how the modified lens flare attributes change with time.

5. Now, map the lens flare object to the created particle object (you can do this by selecting the particle object and then drag and dropping the lens flare effect into the view window).



The final state of flares

Animating Image Effects

Realsoft 3D includes powerful tools for performing post-processing effects on rendered images. The dynamic architecture of the rendering system makes it possible for the rendering engine to output, or make available, any data it computes during the rendering stage. In addition to predefined rendering channels such as Color, Alpha, and Transparency, you can create and define any number of custom channels and use the VSL material system to process them. Thus, it is possible to achieve virtually any post processing effect you can imagine.

Global fog

Tutorial level: Medium

Example project: 'tutorprojects\animation\post images\animated fog'

The fog post effect uses the depth information from the rendering engine and its own attributes such as fog color, saturation, and minimum and maximum distances to compute a global fog.

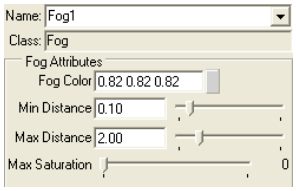
You can animate any of the fog attributes. For example, to create an animation where the saturation of the fog drops from 100% to 0%, do the following:

1. Create a fog object in the select window's Post Image Effects tab by selecting New/Fog. Double click on the new object to open the property window.

2. Turn on animation recording.

3. Move the time slider to the last frame and set the Max Saturation field to 0%. Turn off animation recording.

4. You now have two key frames for the Max Saturation attribute. Move the time slider to see how the fog saturation decreases over time.



Modify the Max Saturation attribute in animation recording mode

If you want to see how the fog looks when rendered, create some geometric objects at different distances in a view window. Next, drag & drop the animated fog object from the select window into the view window. Render the view (with Ray Trace rendering) at several different frames along the timeline to see the effect of decreasing fog saturation.

Rotations

A problem that has existed since the birth of computer graphics is one of interpolating the intermediate frames from given key frames that control rotation. For example, if you have three key frames which represent camera orientations of *front*, *top*, and *back*, you'll typically get a sudden jump as soon as the camera reaches or even passes close to the *top* position. This problem is caused by the fact that the polar coordinate system has singularities at the north and south poles. Rotation curves, which look very smooth and beautiful in the curve editor, are totally incapable of producing nice, smooth rotations in 3D space.

Realsoft 3D solves this problem by using advanced *Quaternion* interpolation for rotations.

The orientation of an object is controlled through three rotation angles called *heading*, *pitching*, and *banking*. Defining any orientation for an object through these three angles is easy and intuitive.

Let's see how well this quaternion system can solve the following well-known rotation problem:

Tutorial level: Medium

Example project: 'animation\rotation\rotating cube'

1. Switch to the front view and create an analytic cube. Activate animation recording.

2. In frame 20, rotate the cube 90 degrees in either direction. To get accurate 90 degree rotation, hold down the Shift key while dragging the rotation handle. This activates the angle grid.

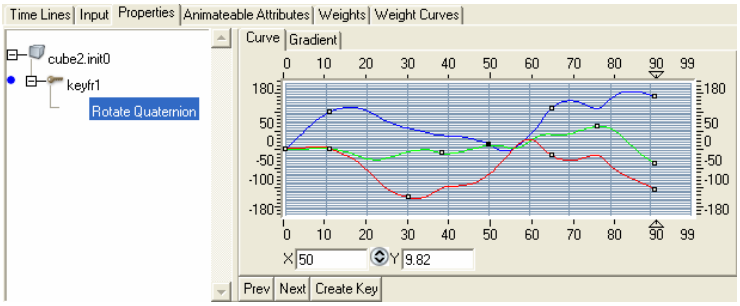
3. Switch to the side view, move to frame 40 and rotate the object 90 degrees.

4. Switch to the top view, move to frame 60 and rotate the object 90 degrees.

Now, reset animation recording and play the animation. As you can see the cube rotates nicely through all defined key frames exactly the way you would expect.

So, it looks like this quaternion interpolation system can handle rotations intuitively. However, there is one minor drawback with this system.

5. Open the choreography window, click the Poperties tab and select the Rotate Quaternion attribute. The three curves that define the heading, pitching and banking angles look very wild and may even contain sudden jumps.



Heading, pitching and banking curves generated by Quaternion interpolation

The explanation for these crazy curves is that the heading, pitching and banking angles simply must jump around to produce a natural looking rotation in 3D space! These strange curves really represent the rotation angles. The problem is that it is difficult to edit these curves in a sensible way.

Euler angles

The quaternion interpolation system produces superior rotations, and is therefore, the default method used. To use the more standard Euler angle interpolation method, check the Rotate in HPB check box in the Gen tab of the property window of your object.



Note

You have to set this option before you animate your object.

Rotating in lattice space

Another way of controlling rotations is to define an axis, and then an angle, which defines how much the object is rotated about the axis. You can do this using the lattice mapping system. The basic idea is the same as for path animations, the difference being that the object is now rotated in lattice space instead of translated.

To illustrate this, we'll create a cube that rotates around a NURBS curve.

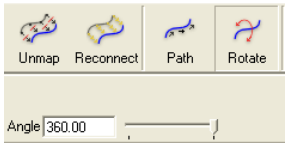
Tutorial level: Medium

Example project: 'tutorprojects\animation\rotation\rotate in lattice space'

1. First create a cube, and then a NURBS curve which defines the rotation axis.

2. Select the cube and the axis (in this order) and activate Lattice Mapping/ Rotate from the toolbar. Specify a rotation angle to the Angle field on the control bar and click Accept.

3. Play the animation and the cube rotates about the axis object.



Set up a rotation in lattice space

One benefit of using this method is that you can now animate the axis, which is a normal geometric object. For example:

4. Activate animation recording and move the time slider to the last frame. Rotate the axis 30 degrees (any direction is fine), and move the axis to a new position.

5. Play the animation. The cube rotates about a moving and rotating axis object.

As an alternative to using the Lattice Mapping/Rotate tool, you can also use the Lattice Mapping/Map tool and then animate the Lattice Rotation attribute manually in the choreography editor.

Texture Morphing

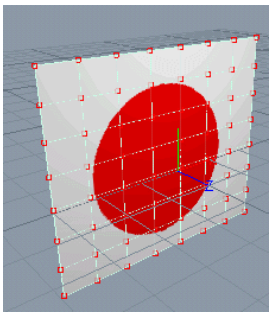
Texture morphing is the process of animating a texture, or an object's material, over time. For example, creating a scene that includes a waterfall, or any type of moving water, would use texture morphing. You have most likely seen movies in which the head of a person suddenly transforms into the head of a terrible monster, or vice versa depending on the movie. Such effects, and more, can be created using texture morphing.

Morphing the shape of a texture

Tutorial level: Medium

Example project: 'tutorprojects/animation/textures/texture morphing'

One way to morph a texture is to change its shape. Realsoft 3D allows you to edit texture control points, just the same way you would edit geometry control points. Since many useful and interesting effects can be achieved this way, let's explore how this is done. This tutorial involves creating a texture that is shaped as a circle and changing it into a rectangle.



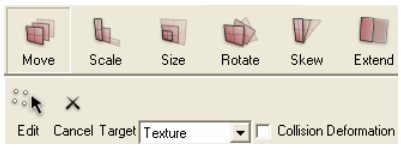
A filled circle painted on a NURBS mesh

1. Create an 8 x 8 NURBS rectangle. This is adequate for this exercise; however, you should note that the more control points the mesh has, the better control you have over texture deformations.

2. Select the mesh. The control bar now shows you the NURBS-specific tools. Click the 'Define UV' tool. This creates custom texture coordinates for the mesh that can be edited and animated.

3. Use the UVimage tool of the Materials tool tab to create a blank texture for the mesh. Turn OpenGL on and use the Circle 3D painting tool to paint a filled circle on the mesh.

You can also use your favorite painting program to create the image, and then attach it to the object using the property window's 'Col' tab.

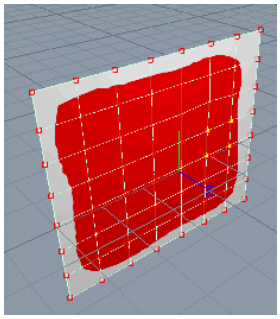


Activate the move tool and set 'Target' to 'Texture'

The next step is to deform the texture coordinates so that the circle turns to a rectangle.

4. Set animation recording on and move the time slider to the last frame.

5. Select the mesh and enter the edit mode. Select the control points whose texture coordinates you want to edit and activate the Move tool. Set the 'Target' option to the value 'Texture'.



Texture coordinates deformed.

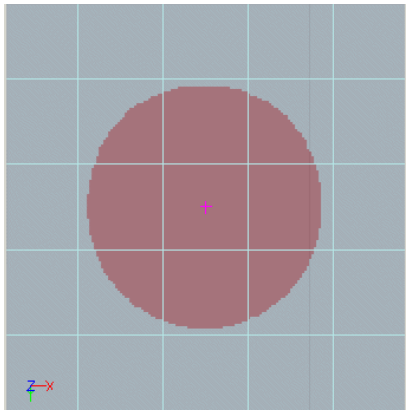
6. Enter two points through the view window to define the movement of the control points. When moving the points, make sure you click the mouse over the actual mesh. Otherwise, the program will not be able to compute how much the texture should be moved on the surface.

7. Continue selecting points and moving them, if necessary, until you have reshaped the texture into a rectangle.

8.When you have finished modifying the necessary texture control points, turn off animation recording and play the animation.

Using the UV Editor

The previous example showed how to directly edit texture object coordinates, and this technique works well for many situations. However, you also can view and edit texture coordinates in the UV Editor. You may find the UV Editor more useful for certain types of texture deformations or adjustments, or it may be that the editor is better suited to your way of viewing the texture editing process. Either way, here is how it is done.



The UV Editor

To use the UV editor to define texture deformations:

1. Open the editor by selecting 'Windows/UV Editor' from the main menu. When you select the mesh, the UV Editor shows you the texture as a backdrop image and renders the texture control points over it.

2. Move the time slider to a desired frame, set animation recording on and edit the control points.

While editing texture coordinates through the UV Editor, view windows show you the result in real time. If OpenGL acceleration hardware is not installed, rendering of large textures may slow down the editing. To avoid this, use non-shaded rendering for the view window.

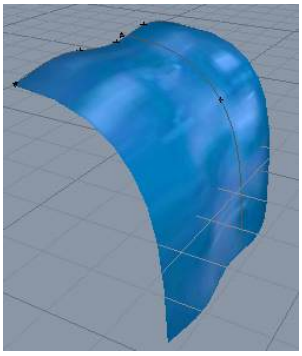
If you now move the time slider, you can see how the texture coordinates in the UV editor change with time.

Creating a waterfall

Tutorial level: Medium

Example project: 'tutorprojects\animation\texture\waterfall'

Let's move on to something more interesting. Next, we will create a simple waterfall animation where a texture that looks like falling water moves along a NURBS mesh.



A textured NURBS mesh representing a waterfall

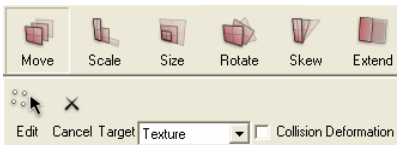
1. Create a textured NURBS mesh that represents a waterfall, see the image below. You can also use the sample project file for this tutorial. Use the property window's 'Col/Texture Map' field to attach the image directly to the surface.

2. Activate animation recording and move the time slider to the last frame.

3. Select the mesh and activate the Move tool. Set the 'Target' option to 'Texture' in the control bar.

4. Now enter two points over the NURBS mesh to define a translation for the texture downward along the mesh.

The texture animation is now ready. Turn off animation recording and move the time slider to see how the texture slides on the surface. In OpenGL rendering mode you can see how the texture moves in real time.



Activate the move tool and set 'Target' to 'Texture'

You can also animate the position of the texture through the property window. In the 'Col' tab you can see four texture transformation fields which allow you to translate, scale, rotate and skew the texture using a numerical interface.

Rendering Animations

Depending on your output requirements, there are many possible ways to render a scene. Realsoft 3D supports a variety of rendering options, including field rendering and motion blur, in addition to the various types of rendering that can be achieved, such as ray tracing, scanline, and outline.

Field rendering

Field rendering is activated from the Animation Settings window by clicking the check box.

When the 'Field Rendering' option is checked, you get a field controller in the animation window that controls the current field.



Click the check box in the Animation Settings window

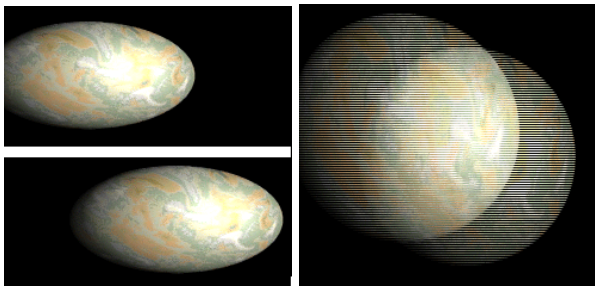
In some cases you may have to control how the fields are rendered and how they are interlaced with each other. The animation settings window contains two controls for this.



Field rendering activated, the Field control appears in the animation window

The first option, 'Shift latter field up', defines whether the latter samples are taken above or below the samples in the former field.

The second check box can be used for defining how the samples are combined.



Even and odd fields on the left. The combined frame on the right.

Motion blur

Motion blur is a rendering effect that makes objects appear to be moving rapidly through the camera's field of view. If you have ever taken a photograph of a moving car at highway speed, you have probably seen the effect.

Animation Controls

Motion Blur is activated from the animation settings window by clicking the check box.

The 'Samples' field allows you to define the number of samples per frame (or field). The higher the value, the better the quality of the motion blur.



Motion blur activated

When motion blur is activated, a new control for samples is added to the animation window.

Using this field, you can go to and render any of the samples in the current frame.



The Sample field shows the current motion sample

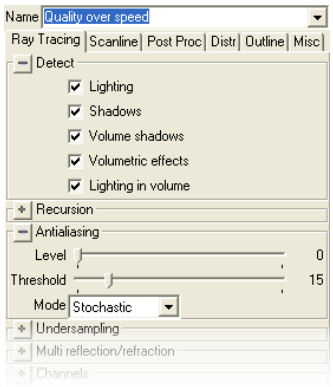
When both motion blur and field rendering are activated, each intermediate image is addressed as follows:

Minute/Second/Frame/Field/Sample

In other words, every field consists of one or more samples. Every frame consists of one or two fields.

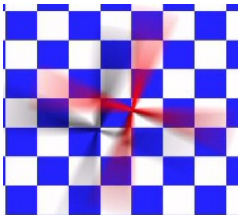
Rendering Settings

When rendering motion blur animations, it is best to use *Stochastic* anti-aliasing (defined in the rendering settings) with the antialiasing level set to its minimum value. Stochastic antialiasing operates over time and antialiases frames well even when using the lowest antialiasing level.



Use Stochastic anti-aliasing method with motion blur

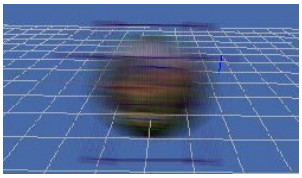
Typically 9 is a reasonable amount of motion samples when rendering animation. Because of the stochastic antialiasing method, the rendering time is, fortunately, not 9 times longer but much less depending on the scene.



A motion blurred image showing a red spinning fan. The fan casts blurred shadows on the floor.

Previewing motion blur using OpenGL

You can use OpenGL to preview field rendering and motion blur animations. To achieve that, make sure OpenGL is enabled on and activate the View Property Window / Render Tab/ Animation Preview / Combine Sub Frames option before starting the preview playback.



OpenGL rendered moving texture mapped sphere

Skeletons

Skeleton is a special purpose deformation lattice object. The idea behind skeletons is quite straightforward: a skeleton is a simple object which is connected to more complex objects. When the skeleton is modified, the attached objects are modified accordingly.

The fact that it is much easier to modify these skeletons than the actual (often quite complex) target objects makes skeletons a powerful animation tool. However, due to generality of the lattice mapping system of Realsoft 3D, skeletons can also be used in various other tasks.

In this tutorial we will demonstrate how to use skeletons to create a walking human body. We will first create a hierarchical human skeleton, then attach the actual model to the skeleton and finally animate the skeleton to take a couple of steps using the key framer.

Creating a skeleton

Tutorial level: Beginner/Medium

Example project: 'tutorprojects\animation\skeleton\body'.

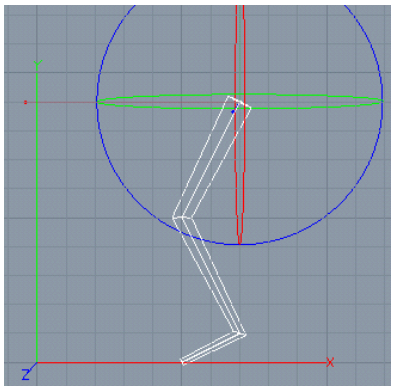
The Skeleton tool creates a skeleton object through the given joint positions. You can also define minimum and maximum constraint angles while drawing the skeleton. Each joint of a skeleton has six angle constraints, which limit rotations of the bones around the joint.

To create a skeleton representing a leg:

1. Activate the Skeleton tool from the Modifier tool tab.

2. Take a side view. Click the first point at the hip bone. Now, move the mouse towards the knee but don't fix the position for the knee yet. We will have to define proper constraints for the thighbone first. Rotate the mouse upwards so that it represents the knee in its uppermost position. Then move the mouse to the opposite direction so that it shows the position of the knee in its back position. Then move the mouse to a position between the two extreme angles and to a suitable distance (the length of a thighbone) and fix the second point (the knee) by clicking the mouse.

3. Then move the mouse towards the ankle and repeat the steps above in order to define constraints for the shinbone as well.



A skeleton representing a simple leg

4. Next define a bone from the ankle to the tip of the toes (we do not create toe bones this time) with appropriate constraints. Then select Accept from the pop-up menu to terminate the skeleton creation.

Once created, skeletons can be modified in various ways.

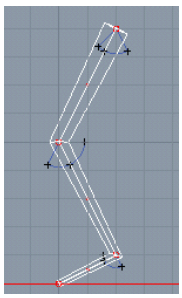
For example, to move a joint of the skeleton created:

1. Select the skeleton and enter the edit mode (as always, you can enter the edit mode either by pressing the space bar, selecting the 'Edit' compass menu, or clicking the 'Edit' tool in the tool control bar).

2. Drag a joint to move it.

While you move the mouse, not only the dragged joint but also all the other joints of the leg become modified. However, bone lengths remain unchanged. This method of modifying skeletons is called 'Inverse Kinematics'.

Inverse Kinematics is the default tool for the joints and therefore it is automatically started when you drag any joint of the skeleton.



Skeleton in the edit state. Drag a joint to apply inverse kinematics.

In addition to inverse kinematics, there are a number of other useful tools for editing skeletons. To apply 'Forward Kinematics' to a joint:

1. Select the desired joint (not the first one).

2. When a skeleton is selected, the tool control bar shows you various skeleton specific tools. Select the 'FK' tool and click the mouse twice in the view window to modify the selected joint angles.



FK tool

To modify the length of a bone:

1. Select a joint

2. Click the 'Length' tool in the tool control bar and enter two points through the view window to adjust the position of the selected joint along the previous bone.

Adding new joints

You can add new joints to a skeleton any time by using the Add Joint tool. For example, let's assume we need to add a new joint to the foot bone representing a toe of the leg:

1. Select the leg and activate the New Joint tool.



Add Joint tool

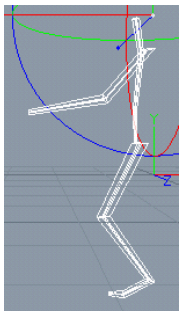
2. Drag the mouse over the skeleton. This outlines a small 'cross mark' which indicates the position of the new joint. When a desired position is found, release the mouse to create the new joint.

Creating hierarchical skeletons

A hierarchical skeleton is a skeleton, which contains one or more child skeletons. A good example of a hierarchical skeleton system is an arm and its five sub skeletons, the fingers. Another example is a spine and its two sub skeletons, the legs.

Next we will create a hierarchical skeleton roughly representing a human body.

1. Switch to the side view (so that you can see the leg from the side). Create three skeletons: a spine, an arm and a leg.



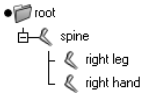
Side view: a spine and two child skeletons



Note

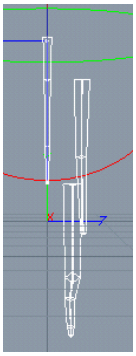
We only have to create the left leg and the left arm for the body!

2. On the select window, drag & drop the hand and the leg skeleton inside the spine skeleton. You should also rename the skeletons so that you can recognize them later.



3. Switch to the front view and move the leg and the hand skeletons to their correct positions.

4. Select both the sub skeletons and click the Attach button in the tool control bar. This attaches the selected skeletons to the nearest joint of the parent (spine) skeleton.



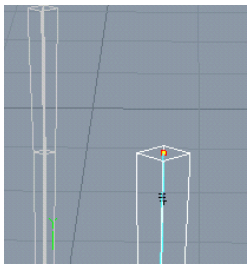
Front view: the leg and arm displaced from the spine

Go back to the side view and try moving joints of the sub skeletons in the edit mode - the inverse kinematics affects the entire body.

Editing angle constraints

If you switch to the front view and try to move one of the hand joints horizontally, you will notice that the entire skeleton moves as a rigid body. Both the arm and the leg refuse to rotate sideways. The constraint angles between the arm and the body do not allow rotations to the side.

There are two ways to edit the constraints. Either you drag the desired constraint angles through the view window, or you use the property window. For example, to modify the constraint of the root joint of the arm skeleton:



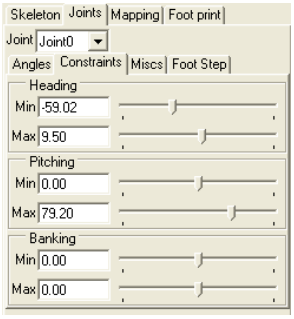
Select the root joint of the arm

1. Select the first joint of the arm skeleton by clicking it

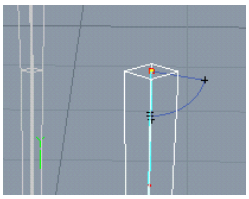
2. Open the property window. Select the 'Spec/Joints/Constraints' tab.

3. As you can see, the 'Pitching' constraint angles are zero. Increase the 'Max' value and decrease the 'Min' value.

While you move the slider, you can see how the 'pitching' handle of the skeleton moves in the view window accordingly. You can also edit the constraint angle by dragging the handle (the cross at the corner of the angle sector) directly through the view window.



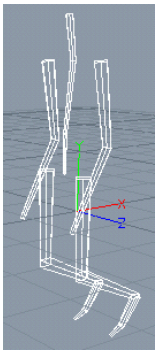
Modify the 'Pitching' angle to allow the hand to rotate sideways



The pitching constraint expanded

Edit also the constraints of the hip joint of the leg skeleton so that it allows appropriate rotations in both the heading and the pitching directions. The knee and elbow constraints are OK - they do not bend sideways.

So, we now have a body, which only has a left leg and arm. The right arm and leg can be created easily. Just select the root skeleton (spine) and apply the DupMir tool.



A human skeleton

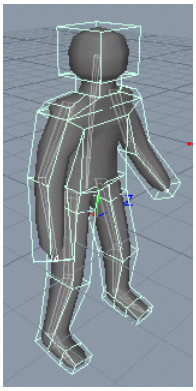


DupMir tool

This duplicates and mirrors all the sub skeletons of the spine. The skeleton is ready.

Attaching bodies to skeletons

The next step is to attach the skeleton to the actual body.



A body and a skeleton

Use the subdivision surface tool set to create a body, which roughly matches the skeleton. If you are familiar with the SDS tool set, you should be able to create a simple body for the skeleton quite rapidly. If you are too busy to model the body, you can load one from:

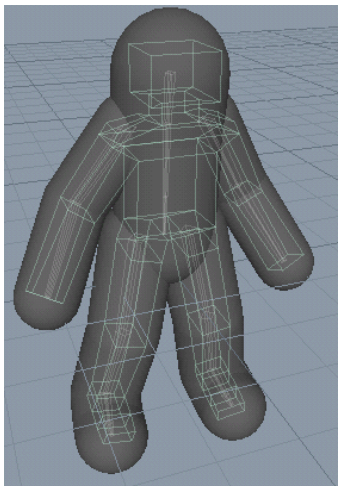
'tutorprojects\animation\skeleton\body and skeleton'

Adjust the skeleton so that it matches the body.

We will attach the body to the skeleton using the 'Multi Map' lattice mapping tool. This tool is capable of attaching a body to a hierarchical skeleton using weighted multi mapping. This means that a single point can be mapped to many skeletons with appropriate weights. The higher the weight, the stronger the skeleton in question affects the point. For example, a point between the start of two fingers can be mapped to both fingers with equal weights, which gives a realistic skin effect.

1. Select the spine and click the 'Map Volume' tool. This turns on visualization of so-called mapping radius property.

2. Click the 'Adj. Rad' tool and enter a point through the view window. If you now move the mouse up/down, you will be able to adjust the mapping volume radius so that those points you want to map to the spine are inside the mapping volume. Repeat the steps for the legs and arms.



Mapping volumes defined for the skeletons.



Note

Shaded OpenGL mode usually shows best what is an appropriate mapping radius. The wireframe of the SDS control polygon outside the mapping volume can be easily detected.

3. Once you have managed to define appropriate mapping volumes, just select the body and the root skeleton (spine) and click the 'Multi Map' lattice mapping tool. Your body is now attached to the skeleton and you can switch the mapping volumes off by selecting the skeletons and clicking the 'Map Volume' button on the tool control bar.



Note

The mapping volumes overlap where the legs and arms join to the body. Correspondingly, points near these joints get mapped to multiple skeletons!



Multi Map tool

To make sure all the points were attached to the skeleton, just modify the skeleton (drag a joint in the edit mode). The body should follow the skeleton.

Tutorial project: 'tutorprojects\animation\skeleton\attached body'



Note

If you defined too small mapping volumes, some points in the model may not get mapped to any skeleton. To fix the problem, select the point, then select the appropriate skeleton and use the standard lattice mapping Map tool with the 'Selected Points' option to attach the point to the skeleton manually.

If the mapping volumes are too large, some points may get mapped to multiple skeletons when they should not. You can detach a point from a skeleton by multi selecting the point and the skeleton and then applying the Unmap tool.

Animating skeletons

Animating skeletons using the key framer does not differ from animating other objects, such as spheres. Just modify the skeleton in the animation recording mode. For example, let's make the skeleton wave its hand.

1. Set animation recording on and jump to frame 20.

2. Switch to the side view and drag the end point of the right arm of the skeleton.

3. Jump to frame 40 and drag the end point of the right arm back to its original position.

4. Switch the animation recording off and play the animation. Tutorial Project: 'tutorprojects\animation\skeleton \say hi'

We have now gone through a basic workflow for creating and animating characters. To keep the example simple, we just attached two sub skeletons to the body to animate legs and arms. However, you can create any number of sub skeletons for controlling the body. Lips, eyes and other parts in your model can be animated this way. In the following chapters, we will go through a number of other useful tools for animating skeletons.

Key framing angles and joint positions

When you modify a skeleton in animation recording mode, the animation system animates the skeleton by key framing the bone angles, regardless of the tool you use to modify the skeleton.

The problem with angle based key framing is that it tends to create 'slipping feet' problems. It is a difficult job to key frame the bone angles so that their motions match exactly the translation of the body.

Instead of key framing bone angles, you can record joint positions. In this case, the system applies inverse kinematics to move the joints through the recorded key positions.

To do this, set the 'Preferred Anim. Method' field in the property window's 'Spec/Skeleton' tab to 'Inverse Kinematics' and then modify the skeleton in the animation recording mode as usual.



Note

When using joint position key framing, key frames are created only for the selected joints (those joints you modify directly with the mouse)!

Mixing angle and position keys

Angle and position key framing can also be mixed. For example, you can first key frame the skeleton using 'Angles'. Then you can define an exact path for a certain joint by key framing its position. This way the system first sets the skeleton to a 'nearly' correct position using angles and then inverse kinematics fine-tunes the position.

To achieve this, use Angles as the preferred animation technique and apply IK Target tool to the desired point. This tells the animation system that the selected joint should be key framed using inverse kinematics and key framed positions.



IK Target tool

Alternatively, you can use Inverse Kinematics as the preferred animation method and use the Angle tool to activate angle key framing for a desired joint.

Anchoring joints

The 'Anchor' tool fixes the joint to object space so that IK cannot move it. This tool is useful when using inverse kinematics to control the joint positions. For example, you can create a moving compass needle and use IK to make it point to the north pole. You have to anchor the origin of the needle so that IK won't pull the entire needle to the North Pole. Let's create a walking leg to demonstrate the 'Anchor' and 'I.K Target' tools.

1. Create a simple skeleton consisting of four joints, representing a leg.

2. Set animation recording on, jump to the last frame and move the entire skeleton horizontally by dragging its 'x' axis. We have now created a simple 'moving leg' animation. Reset animation recording.

3. Enter the edit mode and apply the 'Anchor' tool for the root joint.

4. Select the last joint of the leg and apply the 'IK Target' tool.

Play the animation.

The root joint is now anchored so it is forced to move according to the translation we applied to the entire skeleton. However, the last joint acts as an 'IK Target'. Because we didn't key frame the position of the last joint, it attempts to remain in its original position.

Tutorial project: 'tutorprojects\animation\skeleton\iktarget'

IK stoppers

When you apply inverse kinematics to a joint (say, you pull a finger of a robot), the effect propagates through the entire skeleton and bends and twists it according to joint frictions and constraints.

Let's assume you want to pull the finger so that only the arm is stretched. Once the arm forms a straight line, the rest of the body is 'moved' as a rigid body.

The 'Head' and 'Tail' tools can be used for this. For example, apply the 'Head' tool to the shoulder of a robot. Then you can pull the finger of the hand. Inverse kinematics propagates towards the shoulder and when it reaches the shoulder, it stops and simply moves the joints at the other side of the joint.

 *IK Stopper tools*

Correspondingly, the Tail tool can be used for stopping inverse kinematics from propagating to the tail of the skeleton when one of the parent joints are modified. For example, if you move the robot from its spine, the arm keeps its current position if 'Tail' stopper is set for the shoulder.

Foot steps

A foot step system moves a character along a path defined by a set of footprints. The character's legs are moved using inverse kinematics. Characters with any number of legs can be attached to a footstep sequence. The system assumes only that each leg is either left or right handed.

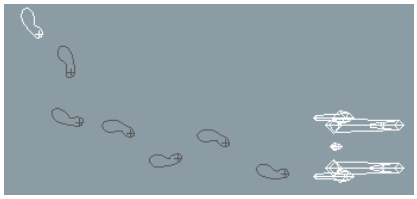
1. Model a skeleton with two legs (or load the skeleton modeled in the previous tutorials, 'tutorprojects/animation/skeleton/body').

2. Switch to the top view (so that you can see the character from top) and activate the Foot Step/Foot tool.

3. Click the mouse to define a set of foot print objects.When a desired number of footprints has been created, select 'Accept'.

This creates a foot print object consisting of a set of left and right handed foot objects.

4. Switch to the side view. You can now see small 'step height' handles above each foot object. Move the skeleton so that its feet are positioned below these 'step height' handles.



Click the mouse on a view window to define a sequence of footprints



Note

It is important that only the joints of the feet are positioned below the line formed by step height handles. The positioning defines which joints are interpreted to be part of the feet - they will be moved using IK.

5. Select the skeleton and the foot step object (in any order) and click Foot Step/Automap tool. This attaches the skeleton into the foot step sequence.

Play the animation and the skeleton walks along the footsteps.

Modifying foot steps

Many ideas used in path animations apply directly to the footsteps too. For example, you can modify the timing by editing 'Lattice Translate' curves in the choreography window.

Each foot object inside the foot step level object defines a 'step height' handle. You can drag these handles in the edit mode to control the step height of the walking skeleton.

You can also modify positions of desired feet to get more interesting animations. Footprint objects inside the footstep object can be duplicated to get longer walking sequences.

Choreography Parametrization

Tutorial level: Intermediate to advanced

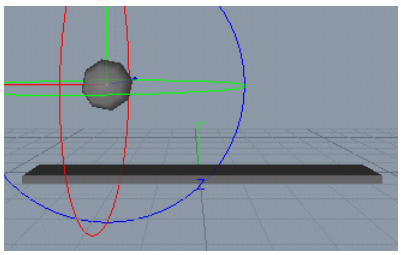
The Realsoft 3D animation system is based on the idea that you can "teach" an object to act based on some input signal. In other animation tutorials, objects were *taught* to act based on time. However, even though all actions occur over time, a time value, such as a frame number, is not the only way or the best way in all cases, to make some action occur in your scene.

This tutorial presents the basic principles for making objects perform some action based on any attribute of any object! This is actually a relatively simple notion - not nearly as complicated as it may sound. When you consider that living organisms in the *real* world respond to and take actions based on numerous types of stimuli (chemicals, light or other forms of radiation, pressure, etc.), it would seem only natural to use these techniques within an animation system.

Although it is true that you could animate anything by setting keys at specific frames, animating certain actions in this way not only can be extremely time-consuming, but prone to subtle synchronization errors. Keying actions based on the attibutes of other objects is an extremely powerful and efficient way to create a wide variety of animation effects, from plants tracking the movement of the sun to a character watching a fly buzzing around the watcher's head.

Tutorial project: 'tutorprojects\animation\input\stretch based on translation'

Let's imagine you want to create a bouncing sphere using keyframes. You would like the sphere to bounce several times during the animation and every time it hits the floor, it should deform as if the collision between the sphere and the floor was causing the deformation.

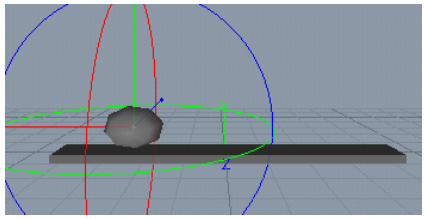


Key 1: The sphere in its initial position.

If you only want to make the sphere to bounce once, there is no problem. You can create a couple of keyframes that control the path and the deformation of the sphere. However, in making the sphere bounce numerous times, it would be great if we could make the sphere stretch automatically every time it hits the floor so that we would only have to worry about the translation of the sphere.

Let's see how this can be accomplished.

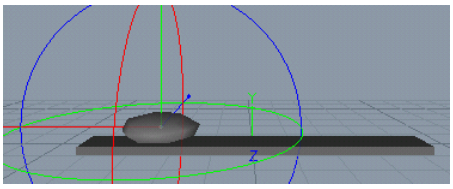
1. Create an analytic cube to represent a tabletop or a floor. Next create an analytic sphere, and in the select window, choose Make Choreographable from the popup menu. Then open the Input tab of the Choreography window. Select the 'sphere.init0' choreograph. Choose New/Keyframer from the popup menu, and then Open from the popup menu to see the created sub choreograph. Select Keyfr1 choreograph. At the right side of the choreography window, change the Input Type to Geometric Object. Now click the Change button, and scroll down the list until you see Translate. Expand the Translate folder if necessary by clicking it and select Translate.y.



Key 2: Position where the sphere touches the floor and starts to stretch.

You are now ready to teach the sphere how it should stretch based on its Y coordinate.

2. Move the sphere downward to the position where the sphere is just about to hit the floor and starts to stretch. Activate the animation recording mode and stretch the sphere slightly in its Y direction. Then reset the animation recording mode.



Key 3: Sphere in its lowest position with maximum stretch

3. Move the sphere further downward to the position where the sphere should reach its most deformed state. Again, set animation recording on and stretch the sphere even more. Reset the animation recording mode.

That's all! Move the sphere up and down using the move tool or by dragging its Y-axis handle and the watch how the sphere stretches every time it hits the floor!

Note that if you followed the previous steps exactly, the sphere should do nothing if you move the time slider. The reason is that you have not taught the sphere to act based on time.

Let's now teach the sphere to move with time. To do this, you create a new time based choreograph, because the current choreograph is sensitive only to the sphere's translation in Y and does not react to time changes in any way.

On the choreography window, select the choreograph sphere.init0. Choose New/Keyframer from the popup menu. This creates a new keyframe choreograph for the object and makes it the current choreograph. By default, this new choreograph is time-based.

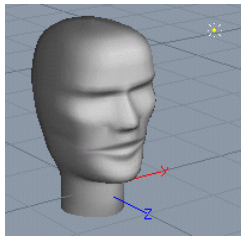
Set animation recording on and keyframe the position of the sphere so that it bounces off the floor. Note that as you move the sphere to a position at the floor, it deforms according to the attribute-based choreograph you just created. When you have keyframed several bounces of the sphere, turn off the animation recording mode and play the animation.

Tutorial project: 'tutorprojects\animation\teaching\stretch based on translation'

A light sensitive fellow

Tutorial project: 'tutorprojects\animation\input\light sensitive head'

Imagine that you want to create a character who does not like bright light. If the light intensity is low, the character is happy. As the light intensity gets stronger, the more unhappy our otherwise friendly character becomes.

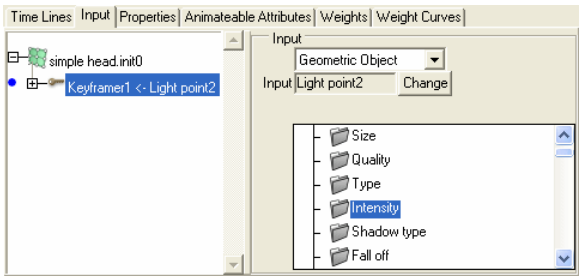


Create a head and a point light source

1. Create a simple head and one point light source. Position the light source so it is in front of the character and slightly above the top of the head. If you are too busy to model a head, you can load in the one from the 'models/nurbs' folder.

2. Select the head and in the select window, choose Make Choreographable from the popup menu.

3. Select the light source, open the choreography window and go to the Input tab. Select the choreograph head.init0. Choose New/Keyframer from the popup menu, then open the head.init0 choreography using the popup menu and click the new keyfr1 choreograph to select it. Change the input type from Time to Geometric Object and then click the Change button. You can now see all the attributes that are specific to a point light source. Select Intensity.



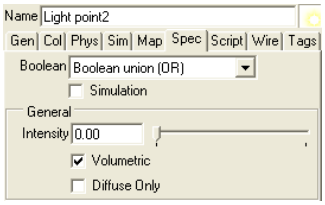
The input of the selected choreograph set to the Intensity of the light source



Note

Instead of selecting Geometric Object and clicking the Change button, you can simply drag and drop the desired geometry object from the select window over the desired choreograph to make it depend on the dropped object.

You have now set up a choreograph that is based on the intensity of the light source. From this point, all that needs to be done is to define the "key" light intensities and modify the facial expressions of the head in the animation recording mode. The only difference between this and recording a time-based choreograph is that you have to adjust the 'intensity slider' instead of the time slider.

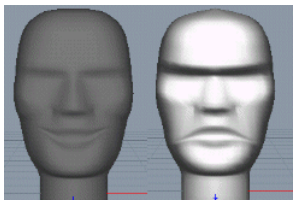


Set the Intensity of the light source to zero

4. Open the property window for the point light. Go to the Spec tab and set the 'Intensity' attribute to a low value (in this case, use zero). Then, set animation recording on and deform the face so that it looks relatively happy. When done, reset animation recording.

5. Now increase the intensity of the light source by moving the slider all the way to the right. Again, enter the animation recording mode. Now edit the character's facial features so that it looks very unhappy. Reset the animation recording mode.

You have now created two 'key' light intensities for the character. Now select the light source and play with the intensity slider in the property window. The stronger the light intensity, the more unhappy the character looks!



The face looks happy under low lighting and unhappy under bright light.

So, we have created a character who does indeed appear sensitive to light. What you actually did was morph the head's facial features as a function of the intensity of the light source. But why would anyone want to define morphing based on the intensity of a light source or some other attribute? There are many reasons, as we will show in the following tutorial examples. Keep reading and be ready for a big surprise.

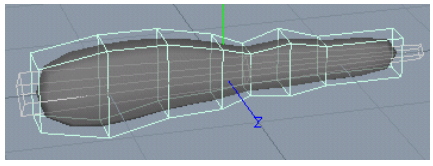
Teaching muscles

Tutorial project: 'tutorprojects\animation\input\bicep'

Imagine that you need to create an arm whose bicep muscle acts based on the angle between the two arm bones, so that the smaller the bone angle, the bigger the bicep. Think about how tedious this task would be if you had to keyframe it based on time. Not only would this be tedious, but extremely limited in application. This is where the real power of attribute based animating is exhibited.

By using attributes rather than time, you can single point edit the bicep at two different bone angles to show the muscle how to deform. In this way, you now have total control over all muscle deformations. Then, you just animate the character's skeleton, making it move as you want, and all the muscles deform automatically according to your definitions.

1. To make such an arm, start by creating and shaping an SDS mesh to be your arm. Then create a skeleton consisting of two bones - make sure the first joint of the skeleton represents the shoulder joint. Remember to define a proper angle constraint for the elbow while you draw the skeleton.



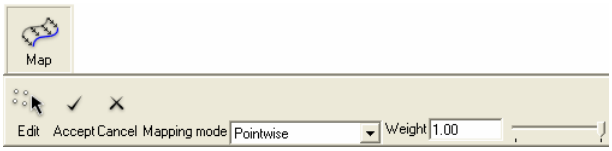
A SDS mesh and a skeleton

Position the skeleton, and then:

2. Select the arm and then the skeleton.

3. Activate the Lattice Mapping->Map tool to map the mesh to the skeleton.

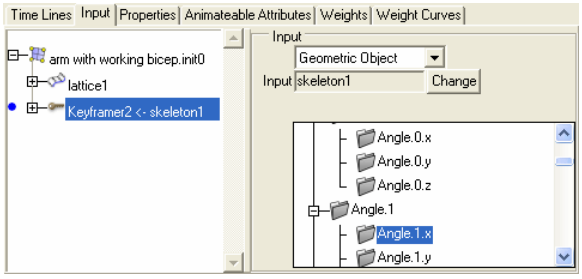
4. Set the mapping mode to the Pointwise option, and click Accept.



Map the mesh on the skeleton using the Map tool and the Pointwise option

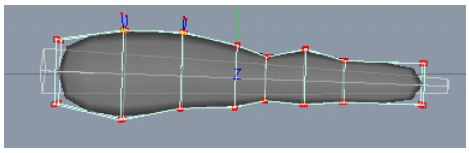
5. Select the skeleton and turn the edit mode on by hitting the spacebar. If you drag the end point of the skeleton so that the arm bends, you can see that bicep area does not react as a real bicep would. Therefore, you need to teach the bicep to get bigger (and contract) based on the bone angle.

6. To do this, open the choreography window to the Input tab. Select the bicep.init0 choreography. Select Open from the popup menu to see its sub choreographs; there is already a lattice item. Choose New/Keyframer from the popup menu, then select the created keyfr1 object. Change the Input type to Geometric Object and select the the skeleton in the select window. Click the Change button. The select the Angle1.x attribute of the skeleton as the input source. Now keyfr1 is a choreograph that is based on the angle of Joint1 of the skeleton.



Set the input of the choreography to skeleton's 'Angle1.x'

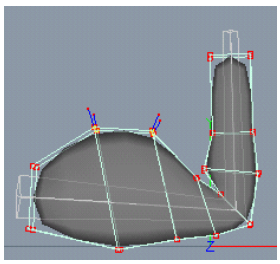
7. Select the skeleton and edit its position so that the arm is straight. Now select the arm mesh and activate the animation recording mode. Use single point editing to make the bicep area look the way you want when the arm is stretched out straight. When finished, deactivate animation recording. Now the muscle knows what to do when the skeleton is opened to its maximum bone angle.



Bicep corresponding the 180 degree bone angle.

8. Select the skeleton and move 'Joint2' so that the forearm is bent at approximately a 90 degree angle. Select the mesh again, and repeat the procedure described above to point edit the bicep area to make it look like the muscle is contracting and bulging. Exit the animation recording mode when done.

That's all you have to do! Now the bicep compresses and stretches based, quite naturally, on the angle between the two arm bones. Move Joint2 and see how the arm looks. There are more refinements you could do to make the arm bend look more natural. If you want to have parts of the arm animate based on other joints or joint angles, use the same technique to create additional choreographs.



Bend the arm and make the bicep bigger

Deformation based on acceleration

Tutorial projects:

- 'tutorprojects\animation\input\deform based on acceleration'
- 'tutorprojects\animation\input\bend based on acceleration'
- 'tutorprojects\animation\input\bend with angular acceleration'

You have probably seen some fancy animations created by advanced simulation systems. If you have used some of these systems, you know that in order to take full advantage of them, you have to know something

about physics. Another problem is that simulations are controlled by a rich set of parameters, and often it is not clear how strong an impact a change to any particular parameter value will have on the final result.

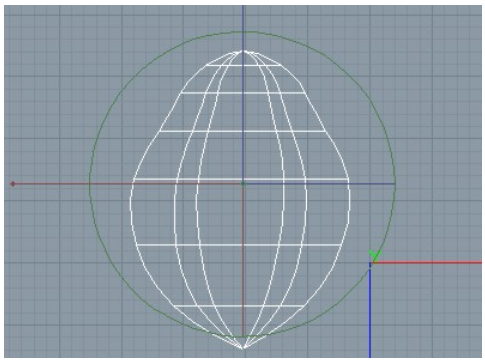
Although Realsoft 3D does include many simulation features, it would, in general, be easier in many cases to keyframe some of these fancier visual effects. Once again, this is what attribute-based choreographs allow you to do.

Imagine that you want to make some character or creature whose body needs to jiggle in a realistic way, based on the body's acceleration (for instance, jumping up and down). In the same way you made the bicep muscle bulge based on the angle of the elbow, you can apply the same principle to make a character's midsection jiggle (or deform in some way) in response to physical properties like the velocity and acceleration of the object.

1. To create this type of animation, start by creating and shaping a cubic NURBS sphere so that it looks similar to the image on the right.

2. Next, from the select window's Geometric Objects tab, right click and choose 'New->Field Evaluator' from the popup menu. This creates a field evaluator object. This object is needed for one reason only: it is capable of evaluating acceleration.

3. Select the field evaluator object and then the sphere object, and then choose Drop to a level from the pop-up menu. Your acceleration-sensitive model is now ready to be animated.



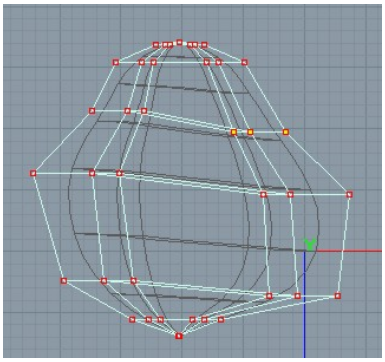
A NURBS sphere reshaped to allow for 'torso' jiggle

4. Select the mesh object, located inside the newly created level object, and choose Make Choreographable from the popup menu. Open the Choreography window to the Input tab. Select the mesh's choreograph and choose New/Keyframer from the popup menu. Open the init choreograph and select the keyfr1 object under it. Then drag & drop the Field Evaluator object from the select window to the choreography list. Choose the Acceleration.y attribute of the field evaluator object.

5. Now go back to the select window. Select the field evaluator object and open the property window to the Spec tab. Set the acceleration field to the following values:

0 10 0

6. Next, edit the model so that its shape is what you want it to look like at this acceleration value: Activate animation recording, select the NURBS sphere and enter the point editing mode. Bend the torso region downward as if it was the high acceleration that deformed the object. Then deactivate animation recording.



Shape of the sphere character when the acceleration value is 0 10 0

7. Since this character will also have to decelerate (that is, respond to negative acceleration), you should also set the character's look under the opposite condition. Change the Field Evaluator object's Acceleration attribute to the following values:

0 -10 0

Activate animation recording, edit the mesh, and then deactivate animation recording.

You have now taught the mesh to 'bend' based on true acceleration. The only thing we need to do now is to animate the position of the level object containing the mesh and the field evaluator object and see how the mesh deforms according to the acceleration.

8. Make sure animation recording is still on, and modify the vertical position of the level object up/down in various time frames. When done, turn off animation recording and play the animation.

Batch Rendering

Realsoft 3D supports number of startup options to support batch rendering. This feature allows you to, for example, render multiple animations over night.

There are three startup options which are useful when using the program as a 'batch renderer'.

-play

Automatically plays the animation using the current file rendering settings.

-firstframexxx

-lastframe yyy

Specify the range to be played. If not given, uses the range specified by the project.

To use batch rendering, you have to select one of the 'rendering specification' objects in the select window (on the last tab) and save the project as usual. Batch rendering uses the selected rendering configuration to render the project.

When -play startup option is given, the program automatically quits as soon as it gets the given animation rendered. This allows you to create script files to render any number of animations. For example, to render two animations you might execute the following script file:

```
realsoft3d -file myproject.r3d -play -firstframe 10 -lastframe 20

realsoft3d -file anotherproject.r3d -play
```



Note

The -firstframe or -lastframe switch the program to 'play range' mode and override any range settings defined by the project. If you don't pass these options, the range settings defined by the project will be used.



Chapter 11. Simulations

Simulations Overview

In the Animation chapter, you learned how to add motion to your scenes using the techniques of keyframing and motion paths, using so-called *traditional* animation tools. Although these techniques can produce impressive results, they still require the animator to define every object movement or attribute change over time. For some types of motion, this can be quite painstaking, and still may not produce results that look realistic. For example, imagine trying to keyframe the motion of a baseball after it has been hit by a batter, or the flight of a golf ball. Even more ordinary, what about a scene where two children throw balls in the air—one is a beach ball and the other a soccer ball. Do you think you could accurately or realistically keyframe the motions of these two balls? Maybe you could, but Realsoft 3D offers tools to make these types of animation much simpler: namely, the *Simulation* tools.

The simulation tools introduce a completely different approach for producing motion. Instead of defining the changes in an object's position over time, you can actually define a "physical" system (simulated of course) that can be used to control an object's motions. Think of some of the physical properties that exist in the real world: there is gravity, there are electromagnetic fields, there is friction, there are forces exerted on objects by wind and water, and there are properties of objects such as their mass, elasticity, rigidity, surface friction, etc.

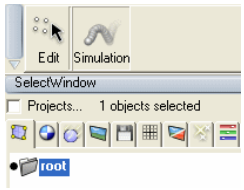
When you use simulation, Realsoft 3D computes all motions automatically, and realistically, based on the laws of physics that produce those motions. Through a combination of setting the Physical and Simulation attributes for objects (Phys and Sim tabs of the Property window), you create the type of *world* in which objects exist; for instance, an Earth-like or Moon-like environment. One advantage is that by using simulation, you can affect (animate) a large number of objects just as easily as a single object.

Our first simulation project is to create an object that emits spheres, which a fan then blows toward a wall. When the spheres hit the wall, they collide and rebound back.

Level: Intermediate

Example project: 'tutorprojects/simulation/firstsimulation'

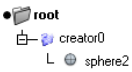
Start a new project. Select the 'root' object and click the 'Simulation' button shown in the tool control bar. This activates simulation for all objects placed under the root level.



Simulation activated for the root level

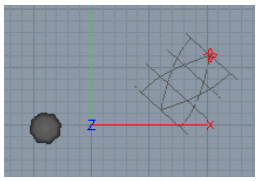
Add a *creator* object to the scene by selecting 'New/Creator' from the Select window's popup menu. Now create an analytic sphere—place it anywhere you like in the view—and drop it inside the creator object. Your hierarchy should look as follows.

Open the Animation window and click the play button. The creator object emits spheres which move in the same direction. This happens, because of the default attribute settings for initial velocity and object lifetime of the creator object.



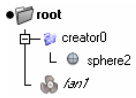
Creator object including one analytic sphere

Click the 'Simulation' tab on the toolbar, and activate the Fan tool. The Fan tool requires that you enter three points in the view window. The first point sets the anchor position of the fan, the second point sets the length of the fan object, and the third point the width. The width does not matter in this example. Enter the points as shown in the following picture to define the fan object.



A fan created

The object hierarchy should now appear as follows:

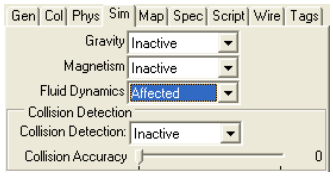


By default, all simulation effects are switched off; therefore, the fan will not yet affect the spheres. To turn the fan on so it affects the spheres, select the sphere object and open the Property window. Go to the 'Sim' (simulation) tab and set the 'Fluid Dynamics' property to 'Affected'. Now all spheres emitted by the creator object are duplicates of this *sample* sphere. All duplicates have the same fluid dynamics properties enabled, and will be affected by the airflow from the fan.



Note

The fan object should affect the spheres via fluid dynamics. We don't want the spheres to generate any fluid flows. Therefore, we set the 'Fluid Dynamics' field to 'Affected'.

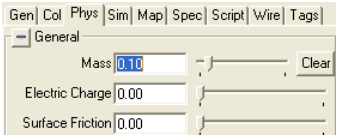


Fluid Dynamics enabled for the sphere used by the Creator object

Now play the animation. If you have followed the instructions exactly to this point, you most likely are not seeing any effect on the spheres; that is, they continue in a straight line as if the fan was not working. This is due to the default values for the mass of the sphere (1.0 kilogram) and the wind velocity (10 meters/second) produced by the fan.

Select the sphere, and on the 'Phys' tab, set its 'Mass' attribute to 0.1.

The default airflow of the fan may also require some adjusting. Select the fan, and on the 'Phys' tab set the 'Fluid Velocity' attribute to 100.0. Now play the animation again. This time, the fan blows the created spheres downward.



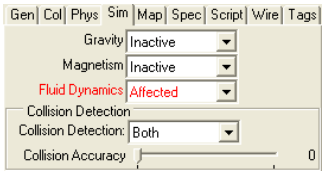
The 'Mass' property set to 0.1 for the sphere.

Note that when the animation reaches the last frame, it is automatically rewound to the first frame. If you cancel the animation before it reaches the last frame, you must manually rewind the animation back to the first frame (use the Rewind button).

The final part of this scene is to create a wall into which the spheres collide. We want the collisions to be detected so that the spheres bounce back after hitting the wall.

Create an analytic cube object, and position it so that it is in the way of the spheres blown by the fan. Play the animation and note where the spheres *disappear*, and place the wall inside of this position. Check from the other orthogonal views to verify that the spheres will hit the cube. If not, adjust the size of the cube.

To make the spheres collide with the wall, you must activate the collision detection effect for both the sphere and the wall. First, multi-select the sphere and the cube, then go to the Property window's Sim tab and set the Collision Detection attribute to 'Both'. (Multi-selecting objects allows you to set one or more attributes to the same value for all selected objects.)



Collision detection enabled for the sphere and the cube

Play the animation and note what happens - the wall is being knocked backward by the collisions with the spheres. There are two reasons for this: first, the mass of the wall is not great enough compared to the mass of the spheres, and second, the wall was set to a Collision Detection value of 'Both', meaning that it is both a *cause* of collisions, and it is *affected* by collisions.

To change the behavior of the wall so that it is not displaced by the collisions, first try setting the wall's Collision Detection attribute to 'Cause'. Play the animation. The wall is no longer moved by the spheres. However, it is still a lightweight object, which cannot bounce the objects that much. This combination makes the wall act strangely in the simulation: the locked position does not match its low mass. Reset wall's collision property back to both, and on the 'Phys' tab, set the mass of the wall to 1000.0. When you play the animation again, you'll note that the wall still moves, but is affected far less by the sphere collisions. This is because we still have a *frictionless* environment; for all practical purposes, the scene we have set up is being treated, physically, as if it were in outer space. To simulate a truly heavy (ground-anchored) wall, change the mass to for example 1000000 kilograms and set the Collision Detection attribute to Cause. Then the wall will not move and has collision behavior of a massive object.

The Collision Detection attribute is set to 'Both' for the sphere. This causes the spheres to collide with both the wall and each other. Change the value for the sphere to 'Affected' and play the animation to note the difference in behavior.

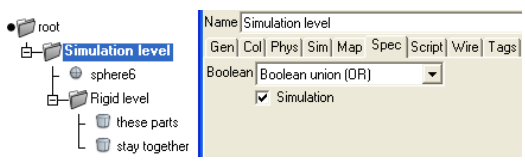
Since you now have a working simulation environment, try testing different values for both the Physical and Simulation attributes of the various objects.

In addition to the attributes available on the 'Phys' and 'Sim' tabs, each object type has *type-specific* attributes, which are located on the 'Spec' tab. For example, the Fan object has a 'Turbulence' attribute for making the airflow turbulent. This is purely a Fan-specific attribute, which is why it is found on the Spec tab.

Simulations and the object hierarchy

The simulation system has an important relationship with the object hierarchy. The 'Simulation' property, which was turned on in the previous example, is *not* a global setting that makes all objects in the scene collide and bounce around. Rather, the Simulation property affects hierarchy levels: that is, *level objects*. The Simulation property controls only the direct sub-objects of the level, and is not propagated into the sub-hierarchies of the level. This situation is shown in the following diagram.

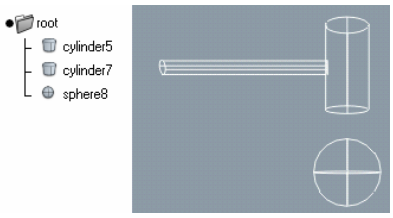
Setting up the Realsoft 3D simulation system to work this way results in an extremely flexible system with very fine-grained control. To demonstrate this, consider a scene where a hammer strikes a ball.



'Simulation' is a level attribute. Simulation will not affect the objects inside the sub-object called 'rigid level'.

Start a new project, and create two analytic cylinders to form a simple hammer, and an analytic sphere for the ball. Set up the scene as shown in the following illustration. Make sure that the hammer head lines up with the sphere in all directions.

Multi-select all three objects, and then choose 'Animations/ Simulations/Collision Detection' from the menu bar. Collision detection is now activated for all three objects. Note that choosing this function also sets the Simulation property on in the parent level of the objects, which in this case is 'root'. To verify what the settings are for the three objects, open the Property window to the 'Sim' tab, and check the value of the Collision Detection property for each objec - note that it is set to 'Both' for each object.



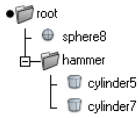
A hammer ready to hit the ball

Now select the two parts of the hammer, click the Simulation tab on the toolbar to open it, and activate the Velocity tool. Draw a vertical line in the view window by placing the first point to the right of the hammer's head approximately at its center, and the second point about one major grid unit below the sphere. This line represents how much the hammer moves in one second.

Now play the animation. When the hammer strikes the ball, the hammer breaks. This happens because the parts of the hammer are treated as individual objects in the simulation, and no force is holding the parts together. To change the behavior, you need to change the object hierarchy.

First, stop animation playback and rewind it to the first frame. In the select window, right click and choose 'New/ Level' from the popup. Rename this new level as 'hammer'.

Select both cylinders and drag & drop them into the "hammer" level. The hierarchy should now look as follows.



A new hierarchy for the simulation

Play the animation. This time nothing happens! This is because the cylinders are now enclosed in a new hierarchy level that has no simulation attributes set yet. We will have to redefine the physical properties for the new level object called 'hammer'.

Select the 'hammer' level and then choose 'Animation/Simulations/Collision Detection' from the main menu. Open the 'Simulation' tab on the toolbar, and activate the Velocity tool. Draw a velocity line like you did previously.

Play the animation. Now the parts of the hammer remain firmly attached, and the ball moves away after being struck. However, note that the hammer is still affected by hitting the ball - it spins as if no one were holding it. Of course, no one is holding the hammer, but what if you wanted it to behave that way? A good question, but that is a topic for later. If you like, experiment with the value of the Collision Detection attribute of the objects, or the mass values of the objects and see how these changes affect the simulation.

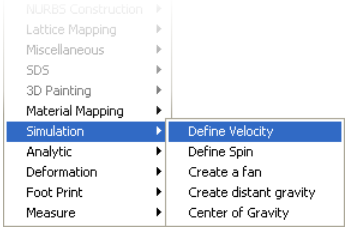
Controlling Velocity and Spin

The simulation system of Realsoft 3D controls objects using two properties: velocity and spin.

Force fields, such as gravity and electro-magnetic fields, modify velocity and spin properties according to the laws of physics. In collisions, velocities and spins change according to rebound energy, mass momentum and other physical properties of the colliding objects so that the objects will rebound correctly.

In addition to force fields, the user can also control the spin and velocity. There is nothing exceptional in these attributes. You can define and even key frame them as you like.

In many situations, it is necessary to define initial velocity and spin for animated objects. For example, when building a bowling simulation, the bowling ball needs an initial velocity.



To define a constant velocity for a sphere:

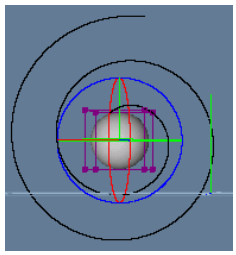
1. Make sure the root level is 'Simulated' and create a sphere.

2. Select 'Tools/Simulation/Define Velocity' from the menu bar.

3. Enter two points through a view window. The two points define the speed for the object - in other words - how much the sphere will move in one second.

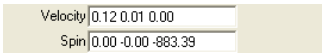
You can also make the sphere spin: select the sphere and the menu 'Tools/Simulation/Define Spin'. Enter three points through the view window: center, start angle and end angle.

The spin tool renders a 'helix' to indicate the angular velocity. The object will rotate this much in one second. (Note - one can set multiple rotations.)



Defining spin

You can also define the velocity and spin attributes numerically. To do this, select the sphere and open the property window. Then go to the 'Phys' tab and enter the desired values to the 'Velocity' and 'Spin' fields.



Defining velocity and spin numerically using the property window

The Creator Object

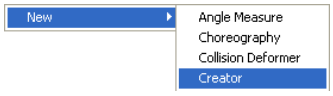
The *Creator* object is a level-object construct that generates duplicate copies of its sub-objects during an animation. However, for the Creator object to work, the level that *contains* the Creator must have the Simulation property turned on. There are many uses for creator objects: bullets being shot from a gun, bees leaving a hive, water dripping from a faucet, just to offer a few examples. Creator objects can be animated using keyframes or paths, just like many other object types.

In this tutorial, we set up a simple simulation where a Creator object throws spheres in all directions.

Level: Intermediate

Sample project file: 'tutorprojects/simulation/emitter'

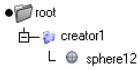
Right click in the Select window and choose 'New/Creator from the popup.



Insert a Creator object

Add a small analytic sphere inside the hierarchy of the Creator object.

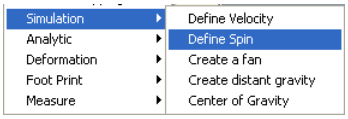
Select the creator object and then choose 'Tools/Simulation/Define Spin' from the main menu to define a spin for the creator level. You need to enter three points through the View window to define the spin. Click once near the left center of the view. Click the second point 5 cm (2 inches) horizontally to the right. Click the third point about 5 cm vertically up from the second point. This creates a counterclockwise spin.



A sphere added into the hierarchy of the Creator object

Select the *root* level and activate the Simulation property for it from the tool control bar. Now play the animation. The creator object emits spheres as it spins. However, note that only the creator object itself is spinning - the spheres it emits do not spin. Now it is time to modify some of the default values.

Select the Creator object, open the Property window's Spec tab, and examine the values of the attributes.

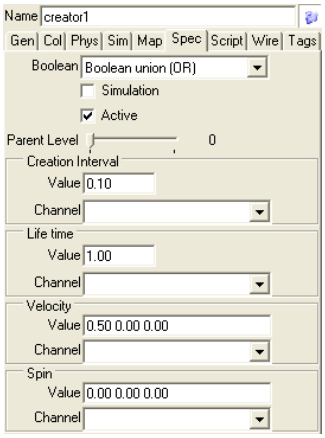


Activate the Define Spin tool, and click three points in the View window to define a spin for the Creator object.

The Parent Level attribute specifies how many hierarchy levels above the creator the emitted objects are inserted. The default is 0, which causes objects to be inserted at the same hierarchy level that contains the Creator object.

The Creation Interval attribute defines the time interval between the creation of emitted objects; that is, it defines how rapidly the Creator emits objects. The smaller the value, the more objects are created. The time interval is specified in the current time units (defined through the Options window). So, for instance, if your time units are frames, setting the creation speed to 15 will cause a sphere to be emitted every 15 frames, or two times per second for an NTSC video system.

The Life Time attribute defines the time interval that an emitted object is active before being destroyed (i.e. its lifespan). The smaller the value, the shorter the object's life. The time interval is specified in the current time units (defined through the Options window). So, for instance, if your time units are frames, setting the Life Time to 15 means a sphere is active for 15 frames (0.5 seconds in NTSC video) before being destroyed.



Attributes of the Creator object

The Velocity attribute defines the rate at which a sphere moves, in meters per second. Since this attribute is a vector value, you define velocities along the three axes. To get motion only along the X axis, you would enter a value such as "0.5 0 0". Velocity is defined in the Creator object's space.

The Spin attribute defines the rate at which an emitted object spins. Since this attribute is a vector value, you define the spin along the three axes.

Set the following attribute values (make sure the Time system is set to 'frames' in the Options window/animation tab):

- Creation Speed = 10
- Life Time = 30
- Velocity = 0.75 0 0
- Spin = 1 1 1

Play the animation. Not bad, but it could still use some fine-tuning. How about adding some gravity to make it more interesting.

Click the Simulation tab on the main toolbar, and activate the Distant Gravity tool. In a View window, enter two points so that you define a vertical line pointing down. You now have a Distant Gravity object in the hierarchy (it should be placed at the same level as the Creator object). Select the Distant Gravity object, open the Property window's Spec tab, and set the Strength attribute to 0.2.

Now select the sphere inside the Creator object level. Open the Property window, and on the Sim tab set Gravity = Affected. On the Phys tab, set the sphere's mass to 0.5. Finally, select the Creator object and on the Spec tab set Creation Speed = 5, Life Time = 90, and Velocity = 0.25 0 0. Play the animation.

Fighter plane shooting bullets

In this tutorial we create a simple FA-18 fighter flying around and shooting.

Level: Advanced

Sample project file: 'Tutorprojects/simulation/fighter'

Model a simple fighter. Make sure all fighter parts are in one level, called 'fighter'.

Insert two Creator objects into the fighter level, one for each wing. Since we are going to use path animation to move the fighter, the guns should naturally move with the fighter.

Create a small sphere that represents a bullet, duplicate it, and move one sphere inside the hierarchy of each Creator object. Position the spheres, so they are in the barrel of a gun; one sphere on the left side left, one on the right.

Select both Creator objects and open the property window's 'Spec' tab. Set the Velocity attribute to 1.0 0 0. In other words, the "bullets" will move at 1.0 m/s (real world bullets move much faster, but this way we have a chance to see them).

You can now play the animation to make sure that the pilot shoots the bullets into the correct direction. If not, either modify the Velocity attribute (try negative values), or simply rotate the creator levels.

Create a NURBS curve to define a motion path for the fighter. Use the 'Tools/Lattice Mapping/Create Path Animation' tool to map the fighter to the path.

Select the parent level of the Creator objects, and activate the Simulation property. This makes the simulation system calculate the velocities of the bullets that are created.

Play the animation. The fighter shoots "bullets" as it flies.

Material controlled creation

Level: Advanced

Sample project file: 'tutorprojects/simulation/bubbles'

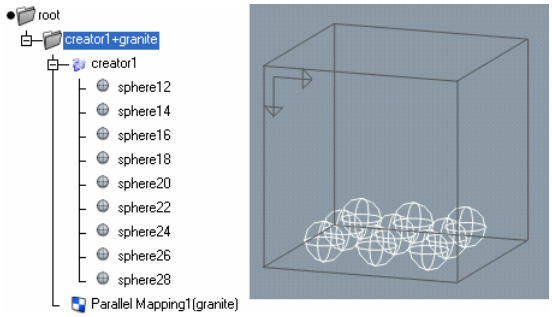
The attributes of the objects which a Creator object duplicates during animation playback can be controlled using materials. The Channel gadget below each attribute in Creator object's 'Spec' tab is used for this purpose.

The following "bubble emitter" example demonstrates how you can use this feature.

First, add a Creator object into the scene. Now create an array of nine spheres, arranged as shown in the following illustration, and place them all inside the Creator object level.

Next, select the Creator object, and drag & drop the "granite" material from the material library onto the View window. The hierarchy should look like that shown above.

With the Creator object selected, open the property window, go to the 'Spec' tab, and set the Velocity to the direction perpendicular to the sphere array (this would be '0 1 0' in the sample scene).

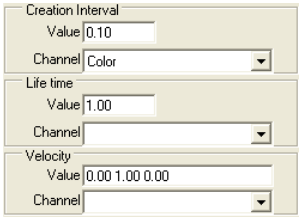


A bubble emitter system

Select Channel = Color from the drop-down list below the Creation Interval attribute. The color defined by the Granite material will define how frequently bubbles appear from each sample sphere. The actual creation frequency is the product of the material channel value and the value set in the Creation Interval field. The latter can be used to scale the values defined by the material.

Select the Parallel Mapping object. Go to the 'Simulation' tab of the main toolbar and activate the Velocity tool. Then draw a vertical velocity line downward.

Select the parent level of the Creator object and activate the Simulation property from the tool control bar.



*Creation Interval is 0.1 * color of a sample object*

Play the animation. The creation frequency is different for each sample sphere. In addition, note that because the granite mapping moves, the creation frequencies also change continuously.

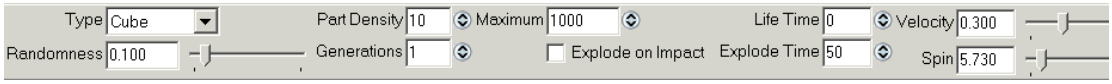
Explosions

In this example we will set up a simple simulation where a sphere explodes after a certain period of time.

1. Create a sphere. Animate it as you like, for example define a velocity value for it.

2. The sphere still selected, go to the Simulation tab of the toolbar and click the Explode tool. A new Explosion modifier appears to the hierachy.

3. The tool control bar shows the properties of the explosion modifier. Set Type to Cubes, Part Density to 10 and Explode Time to frame 50.

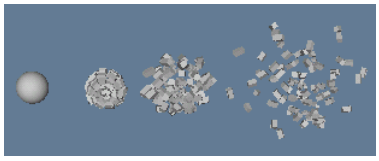


Explosion options

4. Play the animation. Sphere will explode after 50 frames.

This effect breaks the exploding object to a number of pieces. The number of explosion parts can be controlled through the Part Density field.

The Life Time field can be used for controlling how long the exploded parts remain alive.



Exploding sphere

Explode at impact

In addition to a specified time, objects can be exploded by collisions. For example, you can create a gun, which shoots exploding bullets.

To create a simulation where a bullet explodes its target object:

1. Create two spheres: one representing a bullet and one representing a target.

2. Use the Simulation/Velocity tool to define an initial velocity for the bullet so that it will finally hit the target sphere.

3. Select both spheres and activate the Collision tool in the Simulation tooltab. The default options are suitable, so simply click Accept.

4. Select the target and apply the Explode tool.

5. The toolbar shows the option of the Explode modifier. Set Explode Time to zero to switch off time based explosion. Activate the Explode on Impact option. Then play the animation.

Explode at a specified time

Explosion is controlled by an Explode attribute. As soon as this attribute is set, the object explodes. Explode Time and Explode on Impact options simply set this attribute when a certain time is elapsed or when a collision is detected.

The Explode attribute can be animated in a way similar to animating other attributes. For example, if you want a sphere to explode at frame 30:

1. Create a sphere and make it choreographable by applying the menu Make Choreographable from the select window's popup.

2. Open the Choreography window. In the Time Lines tab, select the Sphere.Init0 choreography. Then select New/Keyframer from the popup menu. Open the sub structure of the Init choreography to see the inserted keyframe object. Click the keyframer object to select it.

3. Go to the Animateable Attributes tab. Select the Explode attribute and set the Animated option.

4. In the Properties tab, open the keyframer object and click on the Explode attribute under it. Right click in the graph window and select Set Minimum and maximum values from the popup menu. Make sure that Min x and Max x values match the length of your animation: Min X=0, : Max X = the number of frames in the animation. Make sure that the maximum value for Y = 1.

5. Select Create predefined curve/Constant curve from the popup menu. Add one point to the created curve by dragging the curve. Because the type of the Explode attribute is Boolean, also the type of the created curve is Boolean. Drag the added point to the value 1.0 at frame 30 (X = 30).

6. The sphere selected, create an Explode modifier. Set its Explode Time to zero to eliminate the basic modifier control. A target specific value, defined by the keyframer, will be used instead.

7. Play the animation. Now, when time reaches frame 30, the explode attribute value goes up to 1 and the sphere explodes.

Explode at other events

As you already know, the animation system allows you to animate 'anything as a function of anything'. This is also true for explosions. However, it usually does not make much sense to key frame the explode attribute, because as soon as the explode key frame curve goes up, the object dies taking all the key frame curves with it.

There are numerous cases where the explode option should be controlled by another object attribute. For example, you might want to explode an object as soon as its speed exceeds a certain limit. To achieve this, animate Explode as a function of Speed. Or, you might want to explode an object as soon as its 'y' coordinate reaches a certain upper limit. In this case, animate the Explode attribute by the Translation.Y attribute.

Let's create an animation where an object explodes as soon as it moves below the ground plane, in other words, when its y coordinate gets smaller than zero. The Explode attribute will not be animated in time but by the y coordinate of the object.

1. Create a sphere and apply Make Choreographable to it from the select window's popup.

2. Open the Choreography window and select the sphere's base choreography 'sphere.init0'.

3. Select New/Keyframer from the popup menu of the choreography list. Open the sub structure of the Sphere.init0 choreography and activate the new keyframer choreography.

4. Go to the Input tab of the choreography window. Drag the sphere object from the select window and drop it on the keyframer object in the choreography window. This instructs the choreography to use one of the sphere's attributes for input. Select Center.y from the Attribute list.

We are now ready to animate the Explode attribute as a function of the sphere's y coordinate.

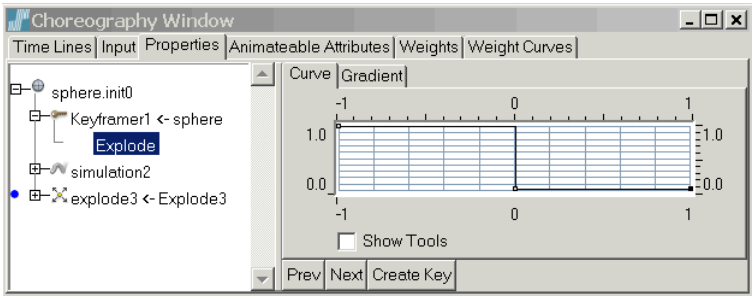
5. Go to the Animateable Attributes tab. Set the Animated option for the Explode property. Then go to the Properties tab and click on the Explode attribute under the Keyframer choreography.

6. Now select Create predefined curve/Constant curve from the curve gadget's popup menu at the right side of the Choreography window. Select Set Minimum and Maximum values from the curve gadget's popup. Define 'Min X' = -1, 'Max X' = 1 (this is the input parameter range i.e. the y coordinate of the sphere center). Define 'Min Y' = 0, 'Max Y' = 1. The vertical Y-axis controls the state of Explode attribute. Check the Rescale box and close the scaling dialog from the OK button. Then define the curve shown below.

7. Use the Simulation/Velocity tool to define an initial velocity for the sphere, making it fall downwards.

8. Add an Explode modifier controlling the sphere. Set its Explode Time to zero.

Play the animation. The sphere moves down and when it reaches the ground plane, it explodes.



Explode is 1 when Center.y is below 0



Note

Explode modifier should usually be the last choreography applied to an object, so that explosion takes all motions and animation details into account. So, if you add new animation effects after setting an object to explode, remember to move explosion to the end of the choreography list.

Gravity

Gravity is always an attractive force, which affects the objects by trying to pull them towards each other. According to Newton's formula:

$$F = g * \frac{m1 * m2}{r^2}$$

Above 'g' is the gravity constant, 'm1' and 'm2' are the masses of the bodies and 'r' is the distance between the bodies. In other words, the higher the mass, the stronger the force. The shorter the distance, the stronger the force. The force is inversely proportional to the square of the distance.

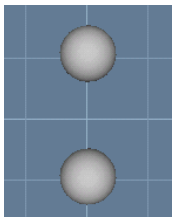
Because the gravity simulation system follows Newton's laws of motion, one can simulate planet systems and other physical phenomena.

Planets orbiting each other

1. Create two spheres: one near the top edge of the view and another one near the bottom edge of the view.

2. Open the Property Window and set the mass for the both spheres to 100000000 kg. The default 1 kg mass is way too light to generate a noticeable gravity field.

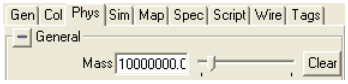
3. Further down on the Phys tab, set the Velocity of the first sphere to '-0.1 0 0' and that of the second sphere to '0.1 0 0'.



Two spheres representing planets

4. Go to the Sim tab and set Gravity to 'Both' for both spheres.

As always, make sure the parent level has its 'Simulation' option set and play the animation.



Set the mass through the 'Phys' tab of the property window



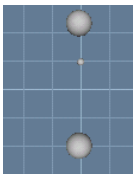
Note

If the velocities of the spheres are too high, the spheres fly away from each other. If velocities are too low, planets approach each other along a spiral orbit and eventually collide.

Example project: 'tutorprojects/simulation/orbiting_planets'

A three planet system

Let's add a moon to the 'orbiting planet' example created above.



A three planet system. The moon orbits the upper sphere.

Example project: 'tutorprojects/simulation/three_planets'

1. Rewind the animation back to frame 0. Move the spheres further apart to make room for the moon.

2. Set the velocity of the moon to [-0.35 0 0]. This will prevent the moon from falling into the upper planet. Set the mass to 100 for the moon.

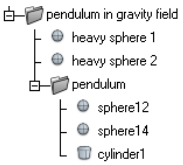
3. Go to the Sim tab and make sure that for all spheres gravity is set to 'Both'.

Play the animation. If you defined the masses and velocities correctly, the moon should orbit the upper sphere. Note how both planets affect the orbit of the moon.

Pendulum

Gravity treats all individual objects, such as spheres, as if their entire mass were collapsed to a single point called 'center of gravity'. Because of this, gravity cannot affect the 'spin' property of single objects. Two orbiting spheres will never start to rotate.

You can define non-homogenous mass distributions by constructing objects from several sub objects. For example, you can create a long pendulum object, which will start to oscillate in space because the gravity pulls its ends with different forces.



The pendulum simulation consists of three objects: two heavy spheres and the pendulum.

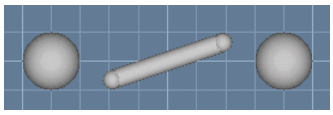
Let's see how this works:

1. Drag & drop a 0.1 meter grid into the view window to activate grid snapping.

2. Create two heavy spheres with equal masses (100000000 kg, for example)

3. Create a level object consisting of three sub objects: two spheres at the ends and a cylinder connecting the spheres. This is our 'pendulum' object. The pendulum object must be positioned exactly in the center of the two heavy spheres.

4. Rotate the pendulum about its center point slightly (say 30 degrees).



The rotated pendulum

5. Select the two heavy spheres and set Gravity to Cause through the Sim tab of the property window. This way the two heavy spheres affect only the pendulum, but they don't attract each other.

6. Set Gravity to Both for the pendulum.

Turn on Simulation option for the parent level and play the animation. The Pendulum starts to oscillate.



Note

If you wish to treat an object, which consists of several parts, as a single entity in gravity computations, just define a mass for the whole composite object using the property window's Mass control. The mass value defined by the level overrules its children's mass values. This naturally speeds up the simulation, because only one single point, the center of gravity, is considered. If

you have set a mass value for the complete level and later wish to restore the use of child-defined mass distribution, click the Clear button behind the mass control.

Example: 'tutorprojects/simulation/pendulum'

Elastic planet

The previous tutorial demonstrated how gravity affects the spin of hierarchical objects. Gravity can also deform objects (a poor astronaut falling into a black hole can turn to spaghetti).

Example project: 'tutorprojects/simulation/elastic_planet'

1. Create an analytical sphere representing a heavy planet. Set the mass of the sphere to 100000000 kilograms.

2. Create another NURBS or SDS sphere (i.e. a freeform object that allows deformations). Go to the 'Phys' tab and define the following physical properties for it:

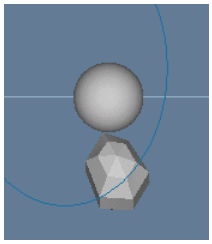
- Velocity: [0.5 0 0]
- Elasticity: 4
- Rigidity: 0
- Rebound Energy: 0.1

As usual, we have to define some initial velocity, which allows the elastic sphere to orbit the heavy sphere. In this case the velocity should be such that it gives a very elliptic orbit for the elastic sphere.

The 'Elasticity' property means that the elastic sphere has internal string forces, which attempt to restore the initial shape. The higher the value, the stronger the string forces are. The result is a 'jelly' like material.

'Rigidity' 0 means that the object can be deformed. For example, warm butter is a good example of a material which can be deformed but which is not elastic. Jelly is an example of an elastic and non-rigid material.

'Rebound Energy' determines how much energy is lost in deformations. If the rebound energy is 1, no energy is lost and the elastic planet would keep jiggling forever. A rebound energy of 0.1 means that jiggling will fade rapidly (due to friction between material particles).



An elastic object deformed by gravity

3. Select both spheres and set 'Gravity' to 'Both' through the 'Sim' tab. Then activate 'Simulation' for the parent level.

Play the animation. The elastic sphere deforms significantly when passing by the heavy sphere.

Birth of a spiral galaxy

Let's create a galaxy, which has a massive black hole in its center.

Example project: 'tutorprojects/simulation/galaxy'

1. Create a small but heavy (100000000 kg) sphere representing the black hole.

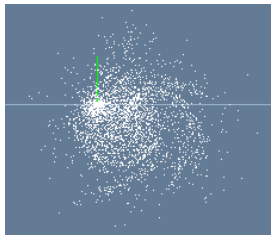
2. Use the particle tool to create lots of stars (1D particles) around the black hole.

3. Set gravity for both objects to 'Both'.

4. Set the following physical properties for the particle object:

- Rigidity: 0
- Spin: [0 0 1].

5. Set the Simulation option for the parent level and play the animation.



A spiral galaxy

The particle object is made completely non-rigid so that gravity can affect each particle as aseparate object. Some initial spin was added for the particles so that they orbit around the black hole instead of falling into it straight away.



Note

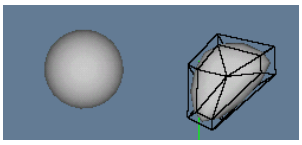
Use a high frame count to preview this animation (say, 1500 frames). After some seconds the particles should form a spiral galaxy. You also may have to increase the Spin value to see the effect.

Collision Detection

The collision detection system of Realsoft 3D can detect collisions between any objects. You are not required to define any type of bounding box for an object - just activate collision detection, define suitable physical properties, and play the animation.

Level**:** Intermediate

Sample project file: 'tutorprojects/simulation/collisions'



Two spheres to be used for a collision

Create an analytic sphere near the left edge of the view and one subdivision sphere in the center of the view. Select the subdivision sphere, enter the point editing mode, and move a couple of points to deform the second sphere.

Select the analytic sphere and choose Tools/Simulation/Define Velocity from the main menu. Enter two points in the View window: enter the first point over the left side of the analytic sphere, and the second point over the right side of the subdivision sphere. The analytic sphere will move the length of this line in one second.

Select both spheres, and then choose Animation/Simulations/Collision Detection from the main menu. This menu function turns on collision detection for the selected objects, and also enables the Simulation property for the parent level.

Now play the animation to see how the spheres bounce after colliding.

Note that collision with the SDS sphere is based on the true surface boundary, not the control polygon. If the shape of a colliding object is irregular and the object moves very fast, it may occasionally penetrate other objects. This can be controlled using the Collision Accuracy attribute (located on the Property window's Sim tab). The default value of zero is usually sufficient. Higher values improve accuracy, but at the cost of animation playback speed. If the Collision Accuracy attribute value is not the same for the two colliding objects, the highest value is used. Try adjusting the value of this attribute and compare the results.

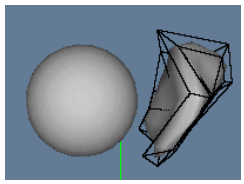
Collision deformations

Level**:** Intermediate

Sample project file: tutorprojects/simulation/elastic_collision

A collision can also deform the colliding objects. Let's make the previous example more interesting by making the SDS sphere elastic.

Select the SDS sphere, open the Property window to the Phys tab, and set the following attribute values:



A SDS sphere deformed by a collision

- Elasticity = 4.0
- Rigidity = 0.1

These attribute settings should make the SDS sphere respond as if it were made of a semi-soft, elastic rubber compound. Play the animation. The collision with the other sphere causes it to jiggle (or oscillate).



Note

Elasticity is a valid attribute only for freeform objects, such as SDS and NURBS objects. Analytic spheres and cubes, for example, cannot be deformed because their geometry does not contain enough points.

Selective collisions

Level**:** Intermediate

Sample project file: tutorprojects/simulation/selective_collision

In the previous two examples, collision detection was activated by selecting the objects and using the Animation/Simulations/Collision Detection function from the main menu. This tool causes collisions to be detected between all of the selected objects. It also sets all the necessary collision attributes for the selected objects. You may have noticed that in the previous examples the Collision Detection attribute was set to the value "Both" for the two spheres, meaning that each object both causes and is affected by collisions.

However, it may not always be the case that you want every object to both cause and be affected by collisions, even though in a realistic universe, this is always the case. One reason to be more selective in setting the value of the Collision Detection attribute is animation playback speed. The computational requirements grow much more rapidly than the number of colliding objects—in fact, the demand is proportional to the square of the object count. For example, if you increase the object count by a factor of 10, the number of object pairs (potentially colliding) grows by a factor of 50.

Manually defining which objects cause collisions and which are only affected by collisions can be used to decrease the computational work for collision detection, often without affecting the realism of the scene. The following example demonstrates this situation.

Create one big analytical sphere (a planet) in the middle of the view window. Then create a dozen smaller analytical spheres (meteorites) around it.

Select the planet sphere and open the Property window. On the Phys tab, set the mass to 1 000 000 000 kg, so that it creates a strong gravity field.

Switch to the Sim tab, and set Gravity=Cause and Collision Detection=Cause.

Multi-select all the meteorite spheres, and set Gravity = Affected and Collision Detection = Affected. Close or minimize the Property window.

Select the parent level of all the spheres, and turn on the Simulation property by clicking the button on the tool control bar.

Set the total frame count in the Animation window to, say, 600 frames and play the animation.

Inertia

Level**:** Intermediate

Sample project file: tutorprojects/simulation/inertia

The Inertia attribute specifies how the mass of an object is distributed inside its volume. If an object's inertia is low, then the mass is located near the center of gravity point; this, in turn causes strong spins during collisions. The higher the inertia, the more the mass is distributed throughout the object's volume, and the stronger the object resists spinning. By default, the program estimates the initial inertia values for an object based on its mass by assuming that the volume is filled with homogeneous matter.

The Inertia attribute is located on the Property window's Phys tab. It is best to use the default inertia as a starting point for experiments. It is easily possible to define inertial values that are physically impossible (at least in this Universe). The result will be that the object exhibits unnatural behavior in collisions, such as wild spinning.

So let's try an experiment with inertial values.

First create three analytical spheres to use as test objects. Stretch two of the spheres so that they become elliptical.

Select the round sphere, open the Property window to the Phys tab, and set the mass to 1 million kg.

Select one of the elliptical objects, and on the Phys tab examine the value of the Inertia attribute. The program sets the default mass of an object to one kilogram; therefore, if the radius of the object is about 10 centimeters, the inertial values will be quite small. Decrease the inertial values along each axis by a factor of 10 by adding a zero after the decimal point. For example, values of 0.05 0.05 0.05 would become 0.005 0.005 0.005. Such values will make this object have an unusually low inertia, making it behave like a heavy lead ball thickly coated with lightweight plastic foam. Make sure that you have defined a suitable number of decimals in the Options Window/Metrics tab.

Select the second elliptical object, and again examine the value of the Inertia attribute. This time increase the value by a factor of ten by removing one zero after each decimal point. For example, values of 0.05 0.05 0.05 become 0.5 0.5 0.5. These values make the second elliptical object have a high inertia, making it behave like a hollow lead sphere having all the mass close to the outer surface.

Close or minimize the Property window, then select all three objects and choose Animation/Simulations/ Collision Detection from the main menu.

Define a suitable velocity for both the elliptical objects using the Velocity tool, located on the Simulation tab. The velocity vectors should be drawn in such a manner that each object will hit the heavy sphere during the animation.

Set the total frame count in the Animation window to, say, 500 frames. Play the animation and compare how each of elliptical objects spins after its collision with the heavy sphere.

After experimenting with Inertia values, you can use the Clear button to the right of the Inertia field to restore the original default values, which usually produces quite suitable and natural spinning.

Surface friction

Level**:** Intermediate

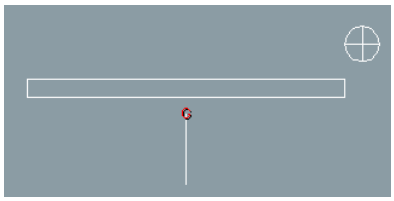
Sample project file: tutorprojects/simulation/surface_friction

An object's surface friction plays an important role in collisions. The following example, a sphere rolling on a table, demonstrates this property. Note that a collision is not always a catastrophic event. A ball rolling on a floor or table is still a collision, albeit, a mild one.

Create a cube to represent a table, and a NURBS sphere slightly above and to the right of the table as shown in the image below. Remember to verify from the side view and top view that the sphere is really aligned with the table.

From the Simulation tab of the toolbar, activate the Distant Gravity tool and draw a vertical line pointing downward. T his defines the direction of gravity, not its strength. The gravity strength value is set on the Spec tab of the Distant Gravity object's property gadget. Set this value to 2.0.

Select the sphere, and from the Simulation tab activate the Velocity tool. Draw a horizontal velocity line from the center of the sphere to about the middle of the table. When you make the sphere affected by gravity, it will fall and collide with the table.



Objects for surface friction simulation

Open the Property window to the Sim tab and define the following properties for the sphere:

- Gravity = Affected
- Collision Detection = Affected

Select the cube object representing the table and define the following properties:

- On the Sim tab, set Collision Detection = Cause
- On the Phys tab, set Mass = 1000000 kg

Although real world tables are not this heavy, they stand on a firm floor, which in turn is anchored to a house, and hence to the Earth. Therefore, the mass resisting the collision is quite large. Note that even though the table will not move when Collision Detection = Cause, it needs a proper mass to create an impact with the sphere.

Select the parent level of your objects, and make sure the Simulation property is on. Play the animation. The sphere should bounce across the table. It may rotate somewhat due to its initial collision with the table and limited computation accuracy. Try also playing with the Collision Accuracy value. Let's see what happens when we give both the table and the ball a non-zero Surface Friction value.

Select both the cube and the sphere. Open the property window to the Phys tab and set Surface Friction = 0.1.

Play the animation again. The sphere stops bouncing rather quickly, and begins to roll. The Surface Friction attribute has the following properties.

- It is defined by the minimum value of the attribute of the two colliding objects. (Even the best tires cannot help a driver on a wet icy road.)
- The value range is from zero to one. At a value of 1.0, no slipping will occur on the surface.

To test this last property of the Surface Friction attribute, set the value for the table and ball to 1.0 and replay the animation.

Rebound energy

Level**:** Intermediate

Sample project file: tutorprojects/simulation/decreasing_rebound

The Rebound Energy attribute defines how much of an object's kinetic energy is lost in each collision. If an object's rebound energy value equals 1.0, this produces a fully elastic collision; that is, the object does not lose any kinetic energy. For example, if a rubber ball hits a concrete wall, it bounces away with exactly the same speed as it had before the collision. If an object's rebound energy value equals zero, this produces a fully non-elastic collision - the object loses all its kinetic energy, such as an egg hitting a concrete wall. When a collision is non-elastic, the majority of kinetic energy is lost (it is converted to heat). If two objects with zero rebound energy collide, they stick together.

To experiment with rebound energy, set up a scene as follows.

Create two NURBS spheres. Multi-select them both, and from the main menu choose Animation/Simulations/Gravity, followed by Animation/Simulations/Collision Detection. Then set the mass of each sphere to 200 million kg.

Play the animation. The spheres should attract each other until they collide, then continue hitting each other and bouncing apart. If the simulation proceeds too slowly, stop the playback, rewind the animation, then increase the mass of both objects. If you look at the attribute values on the Phys tab for the two spheres, you'll see that the Rebound Energy attribute is set to 1.0 (the default). Try setting the Rebound Energy value of both spheres progressively lower (0.75, 0.5, 0.25), and replay the animation. As the value is lowered, rebounding decreases and finally stops.

Electro Magnetism

Electro magnetic force is generated by the electric charge of an object. Unlike gravity, electromagnetic force can be both attractive and repulsive.

A compass is a good example of this force type. The needle points to the North Pole, because the sign of the force field at the North Pole is opposite to the sign of the needle of the compass.

A compass

1. Create two small analytic spheres representing the north and the south poles.

2. Select the north pole sphere and go to the Phys tab in the property window. Set Electric Charge to 10. Then set the electric charge of the south pole to -10.

3. Create a level object consisting of two sub objects representing the two ends of the compass needle. Set electric charge of the 'north' end sub object to -1 and electric charge of the south end to 1.

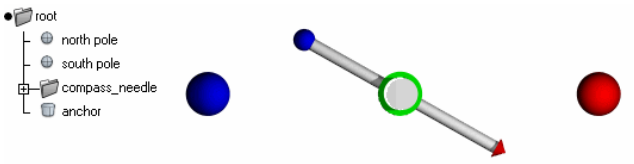
4. Select the 'pole' spheres. In the Sim tab, set Magnetism to Cause.

5. Select the needle level object. In the Sim tab, set Magnetism to Affected. In other words, the needle does not generate a magnetic field, it is just sensitive to the field generated by the north and south poles.

6. Remember to turn on Simulation of the parent level of the compass needle.

In the real world, even tiny compass needles generate a magnetic field. The field is just so weak that most magnetic objects near the compass do not notice it. If you want the needle to generate an electro magnetic field, select Magnetism = Both option. If you have lots of compasses and the field generated by the needle is irrelevant, you might want to switch it off - this simplifies the simulation and increases animation playback speed.

The sample project 'tutorprojects/simulation/compass' includes a slightly more interesting setup: the needle contains two connecting rods and a cylindrical ring in the middle. A 'heavy' pole goes through the ring, anchoring the needle using collision detection. The ring and pole have some friction and their rebound energy is less than 1. This stabilizes oscillations of the needle when it orientates to the magnetic field.



A compass simulation

Fluid Dynamics

With the help of the fluid dynamics simulation tools of Realsoft 3D, you can easily animate various fluid or air flow related phenomena:

- A falling object accelerating slowly towards the terminal velocity
- A fan that blows a bunch of objects away
- Aerodynamic properties of a moving dart turn its peak forward

The first example demonstrates how atmosphere resists rapid motions. Because of this, the velocity of a falling object does not increase forever but approaches the so called terminal velocity.

Tutorial level: Medium

Tutorial example: 'tutorprojects/simulation/falling_objects'

1. Create two analytic spheres, one big and one small.

2. Activate the Distant Gravity tool from the Simulation tab of the toolbar. Draw a vertical line showing the direction of gravity on a view window.

3. Select the two spheres. Open the property window, go to the Sim tab and set Gravity = Affected for the spheres.

4. Select the parent level and turn Simulation on from the tool control bar.

5. Set the frame count to 500 and play the animation.

The natural earth-like gravity accelerates the spheres very rapidly and they disappear from the view window. Stop the playback after about 50 frames. If you look at the Velocity value of the spheres from the property window's Phys tab, you can see that both spheres have exactly the same velocity (which is already quite high, over 20 m/s = 72 km/h). The velocity does not depend on the size of spheres.

6. Rewind the animation.

7. Select the Fan tool from the Simulation tool tab. Draw a fan object with three mouse clicks on a view window: 1. fan position, 2. fan direction, 3. fan radius. The size and orientation of fan is not important this time.

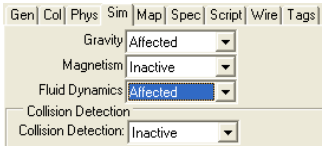


The fan tool selected

8. Go to the Phys tab of the fan's property window. Set Fluid Velocity of the fan to zero (default value is 10 m/s). The reason is that we do not want any wind blowing in this simulation. The fan only adds a non-flowing atmosphere which resists motions.

9. Increase the Density value (it is also in the Phys tab) of the fan from 1.0 to 10.0 kg/cubic meters. The denser the media, the better it resists motions.

10. Select the two spheres. Switch to the Sim tab and set Fluid Dynamics = Affected.



Turn 'Fluid Dynamics' on

11. Play the animation. The spheres start falling again. Now the smaller sphere falls faster. The spheres have an equal default mass of 1 kg. However, the small sphere has a smaller area causing air friction, and therefore it can fall quicker. This is a correct result - a small stone falls quickly, but if you put it inside a balloon it falls slowly.

You may stop the playback again after 50 frames. The property window shows that velocities are now smaller, probably less than 20 m/s. Rewind the animation.

12. As a final step, select the two spheres and decrease the Mass value to 0.1 kg using the Phys tab of the property window. Play the animation again. Now the air resistance is quite significant because of the low weight of the spheres. Both reach their terminal velocity quickly during the first 50 frames. The terminal velocity of the bigger sphere is quite low, less than 5 meters per second.

A funny fan

You may have already noticed that the Fluid Dynamics gadget includes the same list of alternatives as the Collision Detection and force fields: Cause - Affected - Both - Inactive. This means that any object can generate the fluid medium and a flow to it. However, there is one special purpose object available for this purpose, namely the fan object.

A fluid flow generated by an ordinary object, for example a NURBS line, is constant everywhere. A fan can generate non-uniform and turbulent flows. The following example demonstrates this.

Tutorial level: Medium

Example project: 'tutorprojects/simulation/fan'

1. Create a Creator object by using the select window's popup menu New/Creator.

2. Create a small sphere to the left edge of the view window. Open the property window, go to the Sim tab and define Fluid Dynamics = Affected. Even a strong wind cannot move small round objects that weight 1 kilogram (probably made of lead). Therefore, switch to the Phys tab and change Mass to 0.001 (= one gram). Then drop the sphere into the sub hierarchy of the creator object.

3. Activate the Fan tool from the Simulation tool tab. Draw a fan in the middle of the view window. The third mouse click defines the fan diameter; it should be about 10-20 % of the view window's width (see the image below).

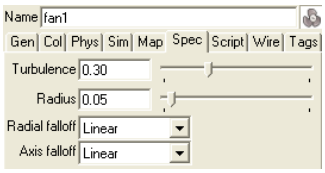


The fan in action

4. Go to the Phys tab of the fan's property window. Set the Spin value of the fan to '4 0 0'. This will make the fan rotate. Increase the Fluid Velocity value to 200 m/s, and Density to 10. This is a truly powerful fan, which can move the spheres easily!

5. Switch to the Spec tab containing the fan specific options. First set Turbulence to 0.3 - this will produce slightly turbulent airflow, like all real world fans do.

6. Both Radial falloff and Axis falloff gadgets are Constant by default. This means that the fan is really strong - it creates a wind that blows through the whole universe. Change both Radial falloff and Axis falloff to Linear. Now the airflow will drop to zero outside the radius of the fan, and get weaker as the distance along the fan axis grows (flow is inversely proportional to the distance). The third alternative Quadric also produces a local airflow, but the falloff is more rapid.



Properties of the fan

7. Close the property window. Turn on Simulation of the parent level and play the animation.

A flying dart

So far, we have used simple symmetric objects for testing fluid dynamics. We have already seen some examples of force fields that spin objects having a suitable sub structure:

- A pendulum in a gravity field
- A compass needle in a electro magnetic force field

By using compound objects consisting of several parts, the same kind of fluid effects can be achieved. The fluid simulation system examines sub hierarchies of objects and each sub component generates its own contribution to the total result. If the mass/area distribution is not symmetric, the result force rotates the object.

The next example, a flying dart, shows this kind of situation. The dart is thrown and it starts falling in a gravity field. The shape of the dart is such that airflow automatically turns the sharp peak of the dart to point to the direction where the dart is moving.

Tutorial level: Medium

Example project: 'tutorprojects/simulation/dart'

1. Model a dart using suitable components. For example, model the two wings at the tail by using two spheres, which are stretched flat in one dimension. The body of the dart can be a cylinder with two spheres at the ends. An analytic cone forms the peak. Enclose all the components in a hierarchy level called 'dart'.



A dart

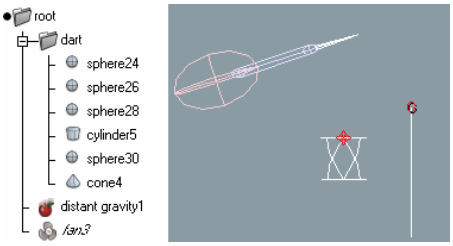
2. Place the dart to the left edge of the view. Direct it to the right and slightly upwards, which is a good starting position for the dart.

3. Go to the Simulation tab of the toolbar. Make sure the dart is selected and start the Velocity tool. Draw a velocity line halfway through the view, to the direction of the dart. The line shows how you 'throw' the dart.

4. Click the Distant Gravity tool of the Simulation tab and draw a gravity line pointing downwards (the length is not important, only the direction).

5. Activate the Fan tool and draw a fan in a view window. Then select the parent level of created objects (root) and turn Simulation on from the tool control bar.

All the required components of the simulation are now included, but we have to adjust some simulation and physical properties. Let's start from the dart itself:



The components of the dart simulation

6. Select the dart and open the property window. Go to the Sim tab. Set Gravity = Affected and Fluid Dynamics = Affected.

Now the simulation already works, but the default properties are quite poor. For example, the default weight of the small dart is several kilograms!

7. Go to the Phys tab. Select the tail components of the dart and set Mass field to 0.001 (one gram). Remember you can set the number of decimals in the Options/Window/Metrics tab from the main menu. Then select the body components and set their mass to 0.003 (three grams). Finally, select the peak and the front part of the body, and set the mass to 0.01 kilograms. If you select the 'dart' level, you can check the total mass, which is automatically computed form the components: it is now '0.02' - '0.03' i.e. some tens of grams. This is quite a realistic value. Note that the center of gravity is in the front part of the dart. If you have tried playing with 'real' darts, you know that they are designed this way.

Now the dart works better. However, in earth's gravity a dart falls very rapidly and hence it must also fly very rapidly to reach its goal. In order to see properly how the dart flies, let's define a 'slow motion' simulation setup.

8. Select the Distant Gravity object from the hierarchy. Go to the Spec tab and decrease the Strength from 9.82 (earth gravity) to 0.1. Now the dart will fall more slowly.

9. Select the Fan and go to the Phys tab. Set Fluid Velocity to zero (no wind).

10. A normal atmosphere does not affect the 'slow motion' dart much. Therefore, increase the Density value from 1 to 10.

You can now close the property window. Set the fame count to 150-250 frames, turn on Simulation for the Root level and play the animation to see the result.



Note

The dart will turn easier if you decrease the default Inertia value. The program estimates the default inertia from the mass and object dimensions. The estimated value can be inaccurate. real dart's mass is heavily concentrated in the front part, which is made of heavy metal, whereas the plastic tail does not weight much. Therefore, the correct inertia value is quite low. For example, divide all the three default inertia components by five.

Scripting in Simulations

You can control various simulation specific attributes by using scripts. This helps to control the behavior of objects acting under influence of force fields.

There are several ways to use scripting in simulations. Firstly, you can attach a script directly to an object. You can also create a special *constraint* scripting object and add it to a hierarchy level in order to control all objects in the same hierarchy level. A constraint object allows you to link objects anywhere in the object hierarchy with each other and control them using scripts.

Attaching a script to an object

You can attach scripts to objects through the property window's Script tab. For example, to control the color of an object using its translation attribute:

1. Create a sphere, set the Command Language in the property window's Script tab to JavaScript and enter the following script:

```
translate = Self.GetTotalTranslate();
Self.SetColor(translate);
```

The script reads the translation attribute of the object and assigns it to the color attribute.

2. Key frame the sphere by moving it in the 'x' direction. Play the animation to see how the color changes when the sphere moves.

Constraining positions

Let's assume you want to keep a sphere falling in a gravity field above the ground plane. Attaching the following script to the sphere does this:

1. Select the sphere and set the Command Language in the property window's Script tab to JavaScript. Enter the following script:

```
center = Self.GetCenter();
if(center.y < 0) {
     center.y = 0;
     self.SetCenter(center);
}
```

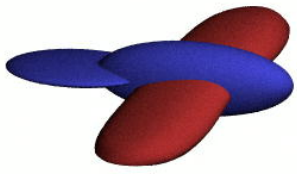
Airplane

The following tutorial example shows how to use a constraint object in a fluid dynamics simulation. We build a little airplane which hopefully manages to take off the ground. This is not an easy task, because designing airplanes is demanding work!

Tutorial level: Advanced

Example project: 'Tutorprojects/Simulation/Plane'

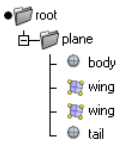
1. Model a simple miniature airplane (about 10 cm long) as shown in the image. It consists of four components: a body, two wings and a tail. Our sample plane consists of two elliptic spheres (body and tail) and two SDS cubes, which have been edited to a wing-like shape.



A simple airplane

2. Select the plane level object. Open the property window, go to the Sim tab and define Gravity = Affected, Fluid Dynamics = Affected.

We will set up a simulation where things happen slower than in a real world. This makes it easier to track the plane when it flies.



3. Select the body and two wings of the plane. Go to the Phys tab of the property window and set their mass to '0.01' (10 grams).

Select the tail and set its mass to '0.002'. The large but light tail will stabilize the plane.

4. Select the body of the plane. Make sure that the Handles/Rotate is set in the view window's popup. Now look at the three move handle lines; the red is object's x-axis, green denotes the y-axis and the blue line is the z-axis. Enter three values to the Fluid Friction field of the Phys tab. The first value is friction in the x-axis direction. If the red line points along the body of the plane, x-friction is very small, 0.01, because the object is most aerodynamic in this direction. If the red axis is not the most aerodynamic axis (but the one between wings or the vertical axis), x friction is slightly larger, 0.03. The other two friction factors are defined the same way. In our sample plane the red axis is the length axis, so the suitable Fluid Friction values are '0.01 0.03 0.03'.

5.Select the wings and define the Fluid Friction the same way as you did for the body. A wing has small fluid friction in two directions, and high friction against the flat wing plane. The orientation of the wings may be different from the body, so check again the red-green-blue orientation of the wing's rotate handle when defining the three fluid friction factors. In the sample plane, the green axis is the vertical axis. Hence the suitable friction factors are '0.01 0.1 0.01'.

6. Select the tail and define Fluid Friction for it ('0.01 0.01 0.1' for the sample plane).

7. Next we define symmetry of airflow around the wings. Asymmetric airflow will create a lifting force. Fluid Symmetry field controls the desired property. A factor '0 0 0' means that air flows around the object in a perfectly symmetric way. In our sample plane, the green object axis is the vertical axis i.e. the direction in which the wings lift the plane. Therefore, we define a Fluid Symmetry value '0 0.7 0' for both wings.

8. Select the complete plane object. Go to the Simulation tab of the toolbar, activate the Velocity tool and draw a short horizontal velocity line for the plane.

9. Activate the Distant Gravity tool and draw a vertical line pointing downwards. This adds gravity to our simulation. The plane has to fight against the gravity to take off the ground. Since our animation happens slowly and in small scale, go to the Spec tab and decrease the Strength field to '0.02'. Now objects fall slowly, and the plane does not have to travel 500 km/h to take off.

10. Activate the Fan tool and draw a fan object in the view window. The fan adds fluid friction to the simulation. To make the simulation interesting, let's use turbulent wind: go to the Spec tab of the property window and set Turbulence of the fan to the full level of 1.0. The direction of wind now changes randomly in space. Go to the Phys tab and decrease Fluid Velocity to 0.2 m/s. Slow enough turbulent wind gives our plane a change to stay in the air for a while. Increase the Density value to 10. Thick air helps to stabilize the plane.

At this point, the plane already tries to take off the ground thanks to its initial velocity. However, it will fall down quickly, because air resists its motion and it has no engine pushing it forward. We will use scripting to add a suitable force, which pushes the plane forward.

11. Activate the Distant Gravity tool again and add another gravity field to the scene. Rename the force object as 'push'.

12. Go to the Spec tab of he property window and adjust the strength of the new gravity object to 0.2 (0.2 N means that the engine is very weak, but the plane weighs only some tens of grams and the actual gravity is also small).

Now the problem is that the new force has a fixed direction. It should always push to the direction of the plane.

13. The push object selected, go to the Script tab of the property window. Set Command Language to JavaScript. Type the following script (remember to hit Enter after the last line):

```
body = Self.FINDBYNAME("body");
center = body.GetCenter();
to = body.GetA();
to.fadd(center);

Self.SetPosition(center);
Self.SetDirection(to);
```

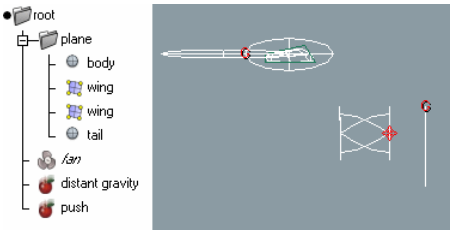


Note

The JavaScript language and the attribute names are case sensitive.

The script above reads the position of the elliptic sphere and writes it to the end point of the push force direction line. Then it sums the 'A' axis of the sphere to the center and assigns to sum to the start point of the force. Note that the orientation of the plane body may be different in your scene: if a green translation handle points the length axis of plane body, then use 'B' instead of 'A' (and 'C' if blue axis shows the right direction). The force orientation may also become opposite to the plane direction, but this can be fixed e.g. by rotating the 'body' ellipsoid 180 degrees (or by changing 'F+' to 'F-' in the script above).

You can also use other attributes of the plane for defining the force direction. The important idea is that the script copies a suitable direction from the plane to the 'push' force. You can see the list of available attributes e.g. by using the 'Input' tab of the choreography window (see the animation documentation for further details).



The plane simulation scene finished

The scene is now ready. Set the frame count to 200-300 frames and play the animation. With some luck, the plane manages to take off and make a beautiful flight. If not, you can decrease gravity and turbidity, increase the size of the tail, balance the plane better, etc. Since the plane has no pilot, it is doomed to crash sooner or later.



Chapter 12. User Interface

Customizing the Toolbar

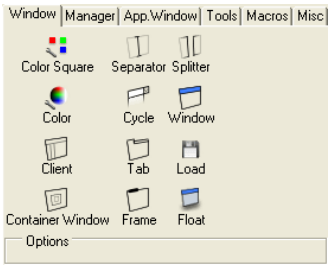
The toolbar in Realsoft 3D is user configurable. You can add new tools into the toolbar, change the order in which the tools appear and delete existing tools.

1. Select the pull down menu Customization/Available Objects Window.

Note that the contents of this window may vary depending on the version and installed modules.

2. Go to the Tools tab and open some tool frames to find the actual tool icons.

3. Drag & drop the desired tools into suitable locations of your toolbar.

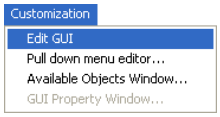


Available Objects Window

Removing tools from the toolbar

You can remove any tool from the toolbar as follows:

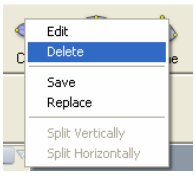
1. Select the pull down menu Customization/Edit GUI.



Activate the user interface editing mode

2. Move the mouse over the desired tool and click the right mouse button to open the pop-up menu. Select Delete from the menu.

3. When ready, exit the GUI editing mode by selecting the menu Customization/Edit GUI again.



Delete a tool from the toolbar

Changing the order of tools

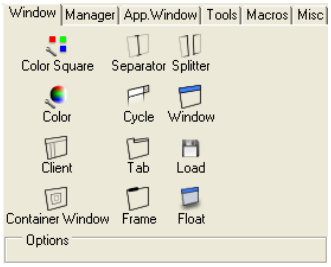
In the GUI editing mode, you can use drag & drop to move a certain tool of the toolbar into another position. However, you can't move or relocate tools away from the toolbar, i.e. tools can exist only in a toolbar.

Customizing the User Interface

Realsoft 3D contains an advanced drag & drop based GUI development system. It allows customization of existing user interfaces, as well as creation of completely new ones.

The 'Customization/Available Objects Window' pull-down menu opens a window, which contains all installed user interface elements.

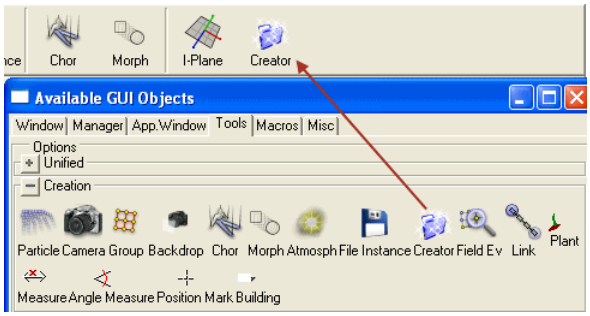
Note that the contents of this window may vary depending on the version and installed plug-in modules.



Available GUI Objects

Adding tools to toolbars

The 'Tools' tab in this window includes all the installed tools. Simply drag & drop the desired tools into your toolbar.

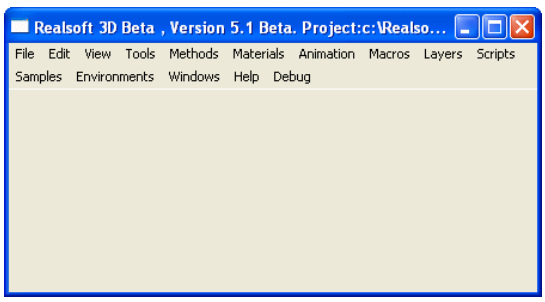


Drag & drop new tools into your application toolbar

Creating new user interfaces

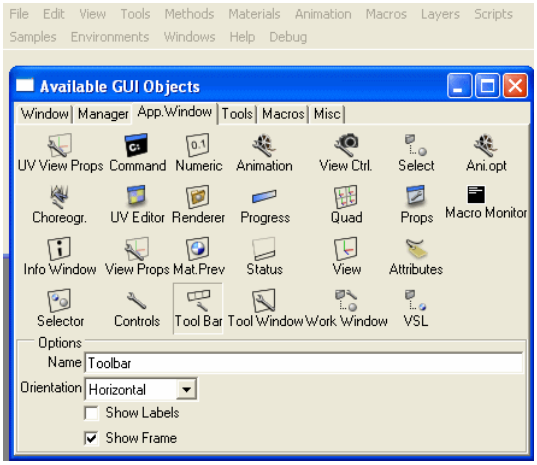
1. Start the program with a new 'blank' project by clicking the 'Blank' project icon of the 'environments' folder, or replace the current environment using the pull down menu 'Environments/Blank'.

2. Select the pull down menu 'Customization/Available Objects Window'. Some of the objects in this window, such as the 'Animation' object, do not define any options and can be directly dropped into the actual main window. Some of the objects, such as the 'Macro' button, require definition of further options, such as the 'macro name', before they can be dragged into the main window.



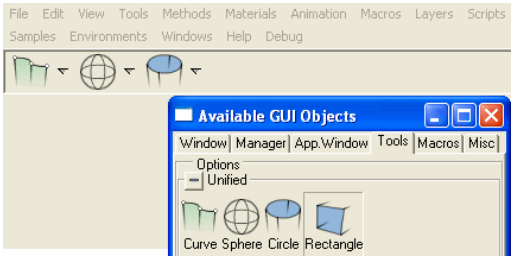
An empty GUI

3. Go to the 'App.Window' tab and click the 'Tool Bar' icon. The lower part of the window now shows you some GUI object specific options. Set the 'Show Frame' option and drag the 'Tool Bar' icon into the 'blank' main window.


Available GUI Objects
Window Manager App.Window Tools Macros Misc
UV View Props Command Numeric Animation View Ctrl. Select Ani.opt
Choreogr. UV Editor Renderer Progress Quad Props Macro Monitor
Info Window View Props Mat.Prev Status View Attributes
Selector Controls Tool Bar Tool Window Work Window VSL
Options
Name Toolbar
Orientation Horizontal
Show Labels
Show Frame

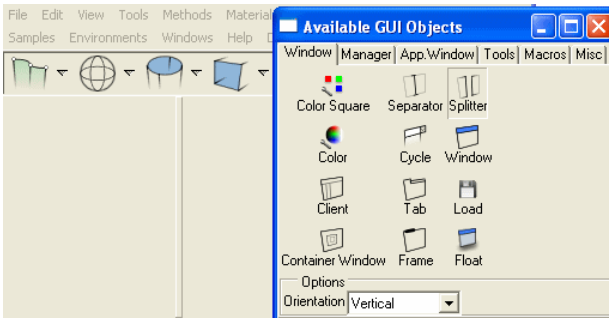
Inserting a toolbar to the GUI

4. Go to the 'Tools' tab of the Available Objects window and open the 'Unified' frame. Drag the 'Sphere' icon into the toolbar. Continue dropping tools until the toolbar contains all the tools you need.



Adding a tool into the toolbar

5. Click the 'Splitter' icon of the 'Windows' tab of the Available Objects window. Again, the lower part of the window is changed to reflect the splitter specific options. Set 'Orientation' to 'Vertical' and drag & drop the splitter into the main window.

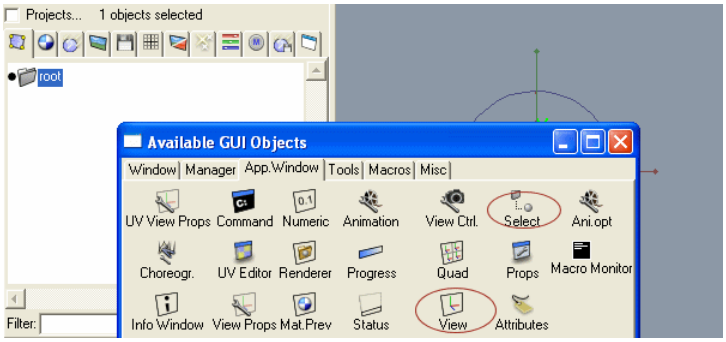


Adding a splitter

6. Go to the App.Window tab and drag the Select object to the left side of the splitter and the View object to the right side of the splitter.

7. When the GUI is ready, close the Available Objects window. To save the created user interface, select the Environments/Save Environment menu and enter a desired name for the file.

You have now created a new fully working user interface and you can use it right away! For example, to create a sphere, simply click the sphere tool.


Projects... 1 objects selected
root
Filter:
Available GUI Objects
Window Manager App.Window Tools Macros Misc
UV View Props Command Numeric Animation View Ctrl. Select Ani.opt
Choreogr. UV Editor Renderer Progress Quad Props Macro Monitor
Info Window View Props Mat.Prev Status View Attributes

Select and View windows added

Geometry managers

When creating new windows, the first thing you have to do is to drop some sort of *geometry manager* into the window. If you do not do this, one is created for you automatically, as soon as the first GUI object is dropped into

the window. Geometry managers are responsible for managing controls in the window, keeping them correctly positioned, aligned and sized according to gadget sizes, the current font and other requirements.

There are currently three geometry managers available:

1. **Packer**. This packs all dropped objects either horizontally or vertically. This is a frequently needed geometry manager.

2. **Row Column**. This arranges its child objects into rows and columns.

3. **Placer**. This manager positions children using relative coordinates.

Once you have selected the desired geometry manager and defined appropriate attributes for it (horizontal/vertical, etc.), you can drag & drop it into the application window.

Then you can drop the actual GUI objects in, and they will be aligned automatically by the geometry manager of the window.

Editing user interfaces

In the previous example we created a customized user interface from scratch. However, we also have to be able to delete GUI objects from the user interface, as well as modify their attributes (such as the orientation of a toolbar).

This can be done in the 'Edit' mode.



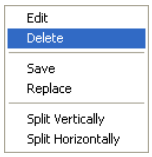
To enter the GUI editing mode, select the pull down menu 'Customization/Edit GUI'. You will no longer be able to use the interface. Instead, you will be able to rearrange the controls in the interface by dragging them.



Drag & drop an icon to a new position

To delete a GUI object from the interface, move the cursor over the object (the frame will be highlighted) and click the right mouse button. This opens a GUI object specific pop-up menu. Select 'Delete' to delete the object.

Select the pull down menu 'Customization/Edit GUI' again to leave the edit mode and return to normal.



Deleting a GUI object

Binding Macros to Keys

The default startup project (startup.r3d) defines a set of default key bindings. For example, when you hit the 'm' key, the move tool is activated.

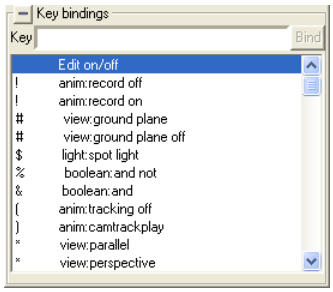
The key binding system of Realsoft 3D implements this behavior. This system contains a list of nodes, each defining two things: a key combination and a macro. Whenever you hit a key, the key binding list is scanned. If a node matching the pressed key is found, the corresponding macro is executed.

Macros can perform various useful tasks. They can activate a tool with the desired options. They can include a sequence of actions. They can rotate the view window, etc.

Defining key bindings using drag & drop

Assume that we want to create lots of NURBS curves using the 'Freehand' option. Activating the curve tool and setting the freehand option is not the fastest way to get the curves created. To solve the problem, you can record a macro, which activates the curve tool with the specified options, and bind the macro to a desired key.

Let's see how this can be achieved.



The 'Key bindings' frame of the select window's Named Macros tab



Note

We suggest that when creating new key shortcuts in the following tutorials, always use the Alt key as a modifier to avoid confusion with the default key bindings. The Alt modifier has been reserved for the user's own shortcuts.

1. Go to the select window's Named Macros tab and open the 'Key Bindings' frame.



Key 'Alt f' entered into the 'Key' field

2. Click the 'Key' field and enter the desired key combination, for example 'Alt f'.

3. Activate the curve tool and set the 'Freehand' option. Then press the Ctrl key down and drag & drop the curve tool icon into the binding list of the 'Key Bindings' frame.

A new named macro with the name of the dragged tool is automatically created and bound to the specified key. Whenever you hit the key, the curve tool with the freehand option becomes activated.

Defining key bindings manually

In addition to the drag & drop interface, you can use a manual method.

For example, to define the 'freehand curve' binding manually:

1. Set 'Macro/Record' on

2. Click the NURBS/Curve tool and set the 'Freehand' option.

3. Reset macro recording and cancel the NURBS/Curve tool.

You have now recorded a macro. You can verify that the macro works as expected by hitting the 'Ctrl + g' key (which executes the current macro). The NURBS/Curve tool becomes active. Cancel the tool after testing that it works as expected.

4. Go to the Named Macros tab of the select window. Open the 'Named Macros' frame. The 'Name' field contains a string 'current'. All new macros are named as 'current' by default. Activate the 'Name' field and rename the recorded macro to a suitable new name, such as 'freehand curve' and press the enter key. The name of the macro is added to the list (alphabetically sorted) of named macros.

The next step is to bind the new named macro to a suitable key. Make sure you don't have any geometric objects selected. If you have, deselect them.

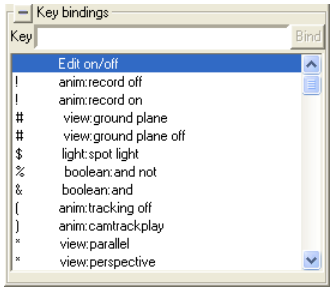
5. Open the 'Key bindings' frame and click the 'Key' field, enter a suitable key combination (in this case the ALT + f key) and then click the 'Bind' button. The key combination is added to the list, which is alphabetically sorted by key command.

This is exactly what happens when defining bindings using drag & drop.

Context sensitivity

Realsoft 3D key binding system supports context sensitivity. This is implemented using *modifiers*. For example, if you hit the 'f' key, a named macro called 'forward one frame' is executed. However, if you hold down the Alt key while hitting the 'f' key, another macro (say 'freehand curve') can be executed.

In addition to the standard keyboard modifiers, such as 'Ctrl' or 'Shift', Realsoft 3D automatically defines modifiers such as 'the type of selected objects' or 'name of the current tool'. This increases the flexibility of the binding system greatly: You can define key bindings, which are executed only when a certain tool is activated or a certain type of object is selected.



Key bindings

You can define a context sensitive macro in the following way:

1. Go to the select window's Named Macros tab and open the 'Key Bindings' frame.

2. Make sure there are no geometric objects selected. If you have activated a tool, cancel it.

3. Activate the Key field in the Key Bindings frame by clicking it with the mouse and hit the Alt and the f key. The Key field shows you '<alt>f'.

4. Now hold down the Ctrl key and hit the 'f' again. The 'Key' field now shows you a string '<ctrl>f'. In other words, <ctrl> is a modifier for the 'f' key.



'Alt f' entered into the Key field

5. In addition to the standard keyboard modifiers, such as Ctrl, Shift and Alt, Realsoft 3D allows you to use any key as a modifier for another key. For example, hold down the 'q' key while hitting the 'f' key. The result is a string '<q>f'.

6. Now activate the Move tool of the toolbar (it's in the Transformation tab), click the 'Key' field and hit the 'f' key. The result is a string '[Move]f'. A macro bound to this key string will be executed by the 'f' key only when the Move tool is activated.

7. Finally, select the 'root' object in the Geometric Objects tab of the Select window and return back to the Named Macros tab. Now click the 'Key' field, hold down the Ctrl key and hit the 'f' key. This produces a combination '[Move][Level]<ctrl>f'. A macro bound to this combination is executed only when the Move tool is activated, a level object is selected, Ctrl is pressed and the 'f' key is hit.

The actual key binding system is simple to describe. When you hit a key, the system attempts to find a key binding, which matches the key string produced by currently set modifiers (such as '<ctrl>f'). If found, the associated named macro is executed.

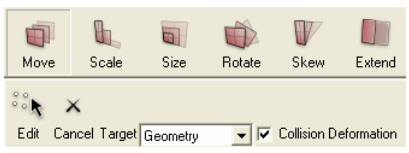
Note that keyboard modifiers are closed inside angle brackets '<>' whereas *current tool* and *selected object* modifiers are enclosed in normal brackets '[]'. Modifiers enclosed with '[]' are optional. If no binding is found, optional modifiers are removed one by one to check matches with more general bindings. For example, if you hit the 'p' key while a 'sphere' object is selected, the binding '[sphere]p' is searched first. If not found, a plain 'p' binding is searched. This may sound complicated, but it allows you to define very powerful and flexible context sensitive key bindings. Let's see how this all works in practice.

Move with collision detection

Let's imagine that you need the Move tool with the 'Collision Detection' option very often, so that you want to define a key binding which allows you to activate the tool efficiently.

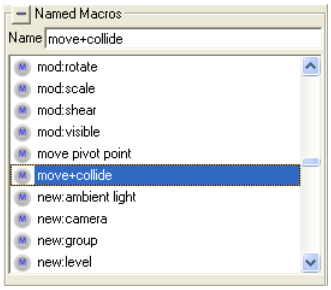
1. Activate macro recording.

2. Click the Move tool and set the 'Collision Deformation' option.



Record a macro, which activates the Move tool and sets its Collision Deformation option

3. Disable macro recording and cancel the Move tool. You have now recorded a macro, which activates the Move tool and sets its Collision Deformation option.

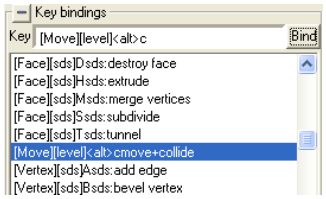


Rename 'current' as 'move +collide'

4. Go to the Named Macros tab in the Select window, open the 'Named Macros' frame and rename the 'current' macro as 'move+collide' and press the Enter key.

5. Make sure you don't have any objects selected (we don't want to create an Object sensitive binding this time). Also, make sure the Move tool is canceled (we don't want to create a Tool sensitive binding). Then click the 'Key' field of the 'Key bindings' frame, press Alt down and hit the 'c' key. Then click the 'Bind' button.

You have now bound the 'move+collide' macro to the '<Alt>c' key, and you should see the string '<Alt>c move+collide' in the key bindings list.



'Move+collide' macro bound to '<alt>c' key combination

6. Create a couple of SDS spheres, select one of them and hit the '<Alt>c' key to activate the move tool with collision deformations.

Tool sensitive bindings

Let's assume that you create curves a lot and you often need to use the 'Closed' option of the NURBS Curve tool. So, let's define a key binding which allows you to activate the 'closed' option for the NURBS curve simply by hitting the key '<Alt>c'.

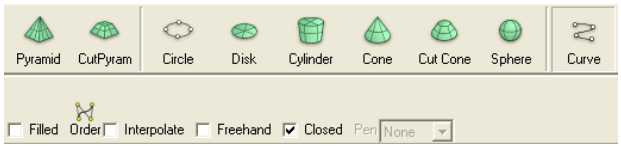


Note

The idea here is that '<Alt>c' should set the 'closed' option only when the curve tool is activated. Otherwise it should activate the move tool with collision detection as defined in the previous example.

1. Activate the NURBS Curve tool. The control bar now shows you the curve specific options.

2. Set macro recording on. Note that macro recording is activated AFTER starting the curve tool.



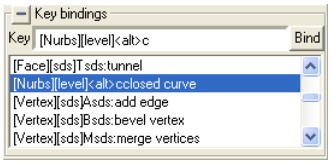
Record a macro which sets the 'Closed' option for the NURBS curve tool

3. Set the 'Closed' check box.

4. End macro recording. You have now recorded a macro, which just sets the 'Closed' option.

5. Go to the Named Macros tab of the Select window. Open the 'Named Macros' frame and rename the recorded current macro as 'closed curve' in the 'Name' field and press the Enter key.

6. Without canceling the NURBS curve tool, open the 'Key bindings' frame and click the 'Key' field. Now enter the combination '<Alt>c'. You should now see the string '*[Nurbs]<alt>c*' string in the 'Key' field. Click the 'Bind' button.



'Closed curve' macro bound to [Nurbs]<alt>c key

Now, cancel the curve tool and activate it again. The closed option is not set by default. Use the '<Alt>c' key combination to get the 'Closed' option set.

Note that the 'Move+collide' binding still works provided that the curve tool is not selected.

Context sensitivity and edit mode

The key binding system is also sensitive to the editing mode of the selected objects. For example, you can bind the 'b' key to the 'Break' tool whenever a NURBS curve is selected and is in the Edit mode. If the selected curve is not in the Edit mode, the 'b' key can execute another tool, for example the Bend tool.

The context sensitivity is implemented by concatenating the 'edit mode' to the key string as a modifier.

Some objects can define several edit modes. For example, SDS objects have three edit modes: point, edge and face. Each mode can have its own key bindings. For example, 's' can activate 'subdivide' in the Face editing mode. In Edge mode, the 's' can activate the 'Split edge' tool.

Another example is a set of bindings to the Tab key, which cycles through all edit modes of an SDS object. This can be achieved by defining the following four bindings:

	[sds]	***Tab***	***sds:edit points***
[Point]	*[sds]*	*Tab*	*sds:edit edges*
[Edge]	*[sds]*	*Tab*	*sds:edit faces*
[Face]	*[sds]*	*Tab*	*sds:object*

In other words, when you hit the 'Tab' key when a SDS object in 'object' mode is selected, the Edit Points mode is entered. If you hit the Tab key again, the macro 'sds:edit edges' is executed, and so on.

Multiple key bindings

You can bind more than one macro to the same key combination. In this case, Realsoft 3D cycles through the macros.

Let's assume you need to switch between smooth and polygonal SDS objects and you would like the '<Alt>p' key to toggle the polygonal option on and off. This is naturally a SDS specific key binding: '<Alt>p' should only toggle the polygonal option when a SDS object is selected.

The default startup project defines these macros already so we don't repeat the recording steps here.

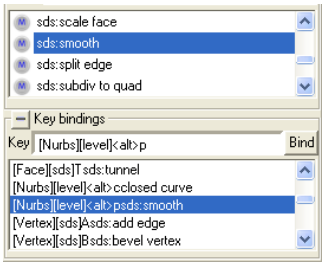
1. Create a SDS sphere and select it (we want to define a SDS specific binding).

2. Go to the Named Macros tab of the Select window and open the 'Named Macros' and the 'Key bindings' frames.

3. Select the 'sds:polygonal' named macro, click the 'Key' field and hit the key 'p' while holding the Alt key down. Then click the 'Bind' button.

4. Select the sds:smooth' named macro and click the 'Bind' button again.

You have now bound two macros to a single key.



Two macros bound to a single key

5. Hit the key '<Alt>p' a couple of times while the SDS sphere is selected. The sphere surface toggles between polygonal and smooth shapes.

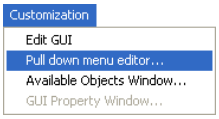
Any number of macros can be bound to a single key this way. The key binding system remembers the last executed binding and next time you hit the same key, the next macro in the macro list is executed.

Saving/loading key bindings

Key bindings are saved with projects. They are included in the Macros section of Realsoft 3D files. You can use the pull down menu File/Save as Startup to save the current project with the modified key bindings to your startup file.

If you want to save only the macros and key bindings, select File/Save As and set the Macros section. This saves both the named macros and the key bindings.

Binding Macros to Pull Down Menus

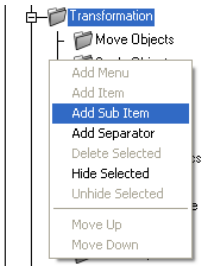


Open the pull-down menu editor

The user can customize the pull down menu strip. Any number of new menu items can be inserted to it. When a menu is selected, the associated macro is executed.

Let's practice this by adding a pull down menu, which activates the Move tool.

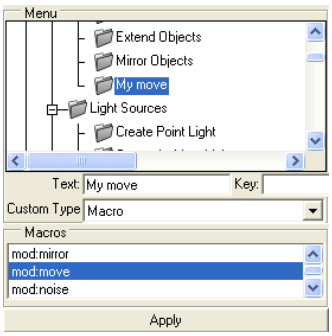
1. Select the pull down menu Environments/ Customization/Pull down menu editor.



Add a new menu item into the Tools/Transformation menu

2. We will insert the new menu item under the Tools/Transformation menu. So, locate and select the Transformation menu item from the opened menu tree view. Select 'Add Sub Item' from the menu editor's popup menu.

This adds a new node 'New Sub Item' into the end of the menu list of the Transformation node.

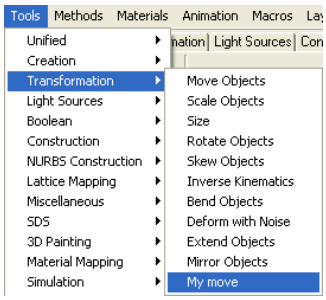


A macro 'mod:move' bound to a 'my move' pull down menu

3. Locate the node and rename it to 'My move' using the 'Text' field. Then select the 'mod:move' macro of the named macro list in the bottom of the menu window. You have now created a new menu item and bound it to a macro.

4. Click the 'Apply' button in the menu window. The specified menu is now inserted to the actual menu strip.

Close the menu editor and try selecting the new menu - the macro 'mod:move' is executed, activating the move tool.



A new pull down menu 'My move' inserted to the Transformation menu



Note

The menu strip is saved with the project. Save the environment to a suitable file to keep the customized menu permanently. (Use the 'File/Save as Startup' pull down menu to save the menu strip to your startup file. Note that this overwrites the startup file.)

Binding Macros to Buttons

Some of the Tools in Realsoft 3D are very general. For example, the Analytic tool creates all kinds of analytical objects.

The good thing is that you can easily find the tool: no matter if you are creating an analytic sphere, a cube, a cut pyramid or a cylinder, just click the analytic tool and the program makes all analytic specific options available for you.

However, if you just want to create certain types of objects, you may prefer a simple button, which selects suitable tool options with one click.

You can construct custom tool palettes using macro buttons, which activate tools with suitable predefined options.

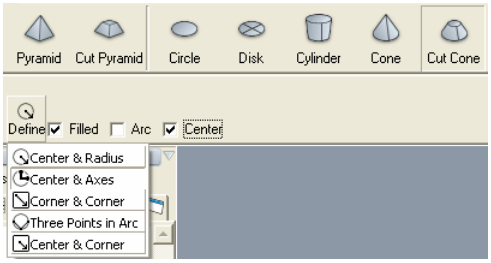
Using drag & drop to create macro buttons

The drag & drop interface provides you with an easy way to insert macro buttons into the user interface.

1. Activate the desired tool and specify desired options.

2. Press the Ctrl key down and drag & drop the tool to a separate window, into another position in the toolbar, etc. to create a 'shortcut' to the tool.

Whenever you click this 'shortcut' button, the original tool is activated with the specified options.



Activate the desired tool with desired tool options

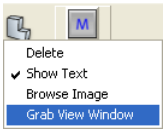
The created macro button uses simple image which shows red 'M' letter. You can find this default image from the 'icons' folder.



Macro button created from the CutCone tool

3. Render nice image for the tool button using the View window and select the Macro button's pop-up menu 'Grab View Window'. This saves the contents of the current view window and attaches the saved image to the macro button.

You may also select any existing image for the macro button by selecting the 'Browse' menu item.



Grab View tool

Creating macro buttons manually

You can drag & drop any number of macro buttons from the Available Objects window into the current user interface. Each macro button can define an icon, a label, a tool tip and a named macro to be executed when the button is clicked.

1. Select the Customization/Available Objects Window pull down menu.

2. Go to the Macros tab of the opened window and click the Macro icon. The macro button specific options are shown.



Macro icon

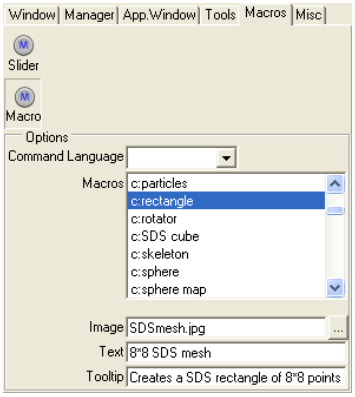
3. Select a macro to be executed from the 'Macro' list. Whenever the button is clicked, the selected macro is executed. You can also Click 'Browse' to locate a desired image for the button. If you don't specify an icon, a default icon will be used. Any image format that Realsoft 3D supports will work (such as JPG, BMP, TGA, GIF, etc.). You can also enter the desired 'Text' and 'Tool tip' strings for the button.

4. When done, drag & drop the macro icon from the Available Objects window into the desired position in the actual GUI (such as a toolbar).



Note

Use File/Save As or Environments/Save environment to save the new customized GUI.



Macro Button specific options defined

Binding Macros to Compass Menus

You can create any number of compass menus.

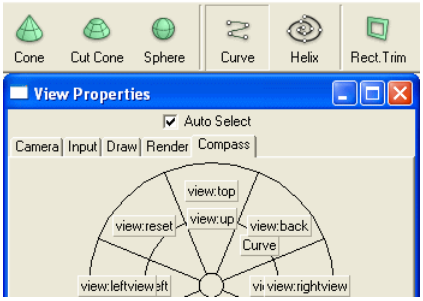
Compass menus are context sensitive in the same manner as key bindings: the contents of the current compass depends on the selected object, current tool and keyboard modifiers. For example, you can create a compass, which is activated when you hold down a certain key while dragging with the right mouse button.

Each menu item in the compass is bound to a named macro, which is executed when the item is selected.

These tutorials show you how to create context sensitive compass menus and how to bind macros to them.

You can easily set up a new compass menu by simply dragging desired tools into it. For example, to add a tool 'create freehand closed NURBS curve' to the default compass menu.

1. Open the view property window and go to the Compass tab. Select the first compass from the Compasses list (Compass).



Select a tool, set specific options and drag & drop the tool into the compass editor

2. Activate the NURBS/Curve tool and set the Closed and Freehand options. Then press the Ctrl key down and drag & drop the Curve tool button into the desired position of the compass menu.

A sphere specific compass

Let's assume you are working in a company designing ball bearings and you frequently need to duplicate a sphere and move it in the 'x' direction.

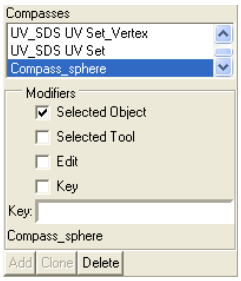
You can increase your productivity by creating a compass menu, which is automatically used when an analytic sphere is selected. It allows you to duplicate and move the selected spheres by using one single operation.

We start by recording a named macro for this purpose.

1. Create an analytic sphere and record a named macro 'mymacro' which duplicates and moves the sphere (If you don't remember how to record macros, check out the macro tutorial in the Modeling chapter).

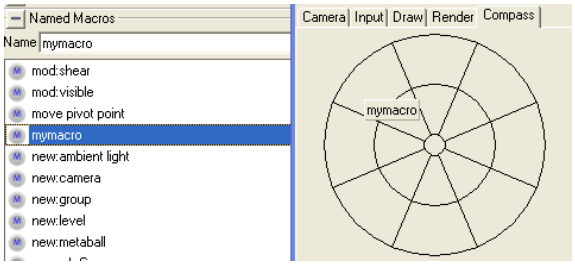
2. Select View Properties from the view's pop-up menu and go into the Compass tab. In this tab you can manage compass menus.

3. Make sure the analytic sphere you created is selected and set the 'Selected Object' check box. Then click the 'Add' button. A new sphere specific compass is created.



A sphere specific compass created

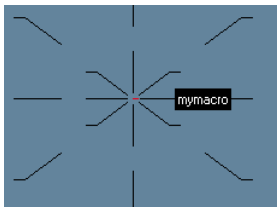
The new compass is shown whenever you drag with the right mouse button and asphere is selected. You can test this in the view window immediately (just drag with the right mouse button in the view window to see the empty compass menu). 4. Drag & drop the recorded 'mymacro' from the 'Named Macros' list of the select window's 'Named Macros' tab into a desired position of the compass.



A compass menu consisting of one menu item

The compass is now ready. You can close the view property window.

5. Select an analytic sphere and drag with the right mouse button in the view window to open the sphere specific compass menu. Select 'mymacro' in the compass menu and the macro is executed.



Select a sphere and drag with the right mouse button in the view window

A key sensitive compass

In addition to selected objects, you can also create keyboard, tool and edit state specific compasses.

For example, you can create a compass, which is opened whenever you drag while holding down the 'F1' function key.

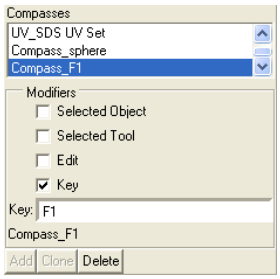
To do this:

1. Go to the 'Compass' tab of the view property window as described in the previous example.

2. Set the 'Key' check box, click the 'Key' string fieldbelow it and hit the 'F1' key. You can now see the string 'F1' in the key field.

3. Click the 'Add' button. This creates a compass called 'Compass_F1'.

4. Drag & drop the suitable macros (or tool icons) to the compass.



F1 Compass created

Deleting and editing compass menus

You can change and delete compass items using the 'Compass' tab of the view property window.

If you want to change a macro binding of a certain compass area, simply drag & drop a new macro into the sector in question.

To delete a named macro, click the right mouse button over the desired compass sector. This opens a pop-up menu, which includes a tool for deleting the macro binding (the actual macro is not deleted).

The Clone and Delete buttons near the bottom edge of the Compass tab duplicate and delete entire compasses.

Localization

The user interface of Realsoft 3D is completely dynamic. Any part of if can be removed, new windows can be inserted.

The dynamic implementation also includes strings and labels associated with graphical controls: such as sliders, buttons and check boxes. All strings can be replaced, translated to another language, abbreviated, etc. and the interface automatically adapts to the new string lengths.

The 'locale' directory contains all strings of the program. The folder includes one file for each Realsoft 3D module. There is a sub folder for each language. The folder named 'english' contains all the built-in strings in the program.

To support new languages, say swedish, find this 'locale' folder in the home directory of Realsoft 3D. Create a new directory under the 'locale' directory. For example 'swedish'.



Note

Setup installs the locale files only if the 'Locale' option is checked. If you can't find the locale \english folder, then rerun the setup. When the installer asks for the installation mode (Compact, Typical, Network etc.) select Custom mode. When the installer asks what components to install, check 'Locale files' and uncheck all other components (assuming that you have already installed the software).

3. Copy the default string files from the 'english' directory into the newly created directory. Translate the strings in the copied .ini files to the new language. Translate only the strings after the '=' character on each line. The strings before the '=' character are used for identifying the strings and should not be changed.

4. Open the StartOpts utility program and set language to the new language, say swedish.



Note

The string files in the 'locale\english' directory are automatically generated from the source code and should not be modified.

You can also select the localization with a startup parameter as follows:

```
realsoft3d.exe -lang swedish
```

Localizing user defined strings

In addition to built-in strings, the end user may create new user interfaces, such as windows and tool bars. Strings in these user defined display elements can be translated as well.

Localizing Tab labels

Tab labels can be defined in the r3wid2.ini, under class [r3guitab].

For example:

```
[r3guitab]

SDS=Subdivision
```

will change the 'SDS tool tab to 'Subdivision'.

Localizing Window names

Window names (strings you see in the Windows pull-down menu), can be localized in r3wid2.ini, under the class [r3drfwto].

For example, to change the 'Property Window' to 'Properties':

```
[r3drfwto]

Property Window=Properties
```

Localizing Docked Window names

Docked windows, such as 'Advanced Search' or 'Animation Settings' can be localized through the r3wid.ini file, under the class [r3clnwin].

For example, to change the 'Animation Settings' title to 'Anim. Options':

[r3clnwin]

Animation Settings=Anim. Options



Chapter 13. Scripting

Introduction to Scripting Languages

Scripting languages (or macro languages) can be used for the following tasks:

- Create and edit geometric and other objects through a command line interface.
- Expand and customize the user interface. For example, you can create a slider, which controls the bone angle of a skeleton.
- Write custom material effects for photorealistic rendering.
- Define *constraints* between objects. For example, the position of an object can be defined as a function of two other objects.
- Create intelligent behavioral animations in which procedures written in macro languages control how objects behave.
- Expression evaluation. The numeric interface for creating and modifying objects, for example, allows you to use expressions to define properties for objects. When creating a sphere, you can fetch information from other objects using expressions like (*sphere1.center.x* + *sphere2.center.y)/2.0*.

The structure of Realsoft 3D is 'object oriented'. Correspondingly, scripts written by the user are always associated with some sort of an object.

Controlling Geometric Objects

Let's create a sphere whose position in the 'y' direction is fixed to 0.0. In other words, the sphere is not capable of moving in the up/down direction.

1. Create an analytic sphere.

2. Select the sphere, open the property window and go to the Script tab.

3. Select JavaScript from the Command Language control and enter the following text into the text field:

```
center=Self.GetCenter();
center.y = 0;
Self.SetCenter(center);
```

Now, try to move the sphere with the mouse. You can move it horizontally but not up or down!

Let's create a simple animation. Try the following script:

```
center=Self.GetCenter();
center.y = Time;
Self.SetCenter(center);
```

On other words, the 'y' position of the sphere depends on the time. Play the animation to verify this.

Binding Scripts to GUI Objects

Let's assume you have to create animations so that there is a key frame in every 5fth frame.

1. Open the Environments/Customization/Available Objects window.

2. Go to the Macros tab. Click the Macro button. The lower part of the window shows you now the macro button specific options.

3. Set the Command Language field to JavaScript. You can now enter the desired program into the text control. Enter the following text:

```
layerlist = GetJS("");
currentproject = layerlist.GetCurrentLayer();
animator = currentproject.GetAnimator();
frame = animator.GetCurrentFrame();
frame = frame + 5;
animator.SetCurrentFrame(frame);
```

4. Enter an appropriate name and a tool tip for the button (such as 'Jump 5'). You might also want to paint a nice icon for the button using your favorite painting program. Then select it using the Image browse button of the macro button controls.

5. Drag & drop the macro button icon into the toolbar.

Every time you click the macro button, the animation time is increased by 5 frames.

JavaScript

Introduction

JavaScript provides you with a sophisticated, object oriented scripting interface to Realsoft 3D.

These tutorials describe how to use JavaScript to access Realsoft 3D functionality. They are intended for programmers who are already experienced at using JavaScript or some other object oriented programming language.

JavaScript is linked to Realsoft 3D through set of wrapper objects. There is a JavaScript object for each Realsoft 3D object, allowing you to access the functionality of Realsoft 3D through native JavaScript interface.

It should be emphasized that JavaScript interface was designed to provide you with total control over Realsoft 3D. It allows you to customize and expand the functionality of the program. Therefore, the structure of the JavaScript interface corresponds to the structure of Realsoft 3D application, as defined in the Realsoft 3D Software Development Tool Kit.

It is also important to understand that JavaScript objects and Realsoft 3D objects are not the same thing. Realsoft 3D objects can exist with or without JavaScript.

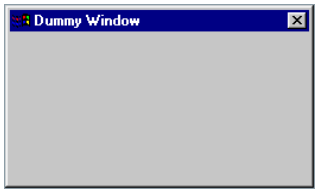
Creating User Interface Objects

Creating a window

You can create a window by calling:

```
include("oops/r3window.js");
win = new r3Window(R3WGA_Parent, _r3gui)";
win.REALIZE();
```

This creates an empty window with default size, position and other attributes.



Empty floating window

The first call.

```
include("oops/r3window.js");
```

loads in JavaScript interface to Realsoft 3D Window objects, if not already loaded.

The second call:

```
win = new r3Window(R3WGA_Parent, _r3gui);
```

creates a window object. Parameters for the construction function specify properties for the window to be created. Each property is defined by a tag-value pair. For example:

```
R3WGA_Parent, _r3gui
```

defines the parent window.

All properties for the r3Window class can be found in 'scripts/js/oops/r3window.js header file, or one of the files from which the r3window object is derived.

There are two ways for setting desired attributes: by passing them to the constructor function as a tag list, or by setting them by calling the appropriate Set method.

For example, r3window.js defines attribute id 'R3WA_Title'. This attribute can be set by calling the 'SetTitle' function (just replace the 'R3WA_' prefix with 'Set' to get the name of the function.

```
include("oops/r3window.js");

win = new r3Window(R3WGA_Parent, _r3gui);
win.SetTitle("My Window");
win.REALIZE();
```

Another way to create a window with desired title is by passing the attribute id, with associated value to the constructor function:

```
include("oops/r3window.js");

win = new r3Window(R3WGA_Parent, _r3gui, R3WA_Title, "My Window");
win.REALIZE();
```



Note

Realsoft 3D defines variable '_r3gui'. This variable refers to the Realsoft 3D main window. The R3WGA_Parent tag must always be given to the constructor function because it is not possible to create a window without specifying the parent for it. This is true of all other user interface specific objects, such as sliders and buttons, to name a few.

The third call:

```
win.REALIZE();
```

makes the window visible to the user. You have to call this method whenever you create a new window object. This is also true for all objects derived from the r3Window object.

Geometric managers

Realsoft 3D provides you with an advanced toolset for designing user interfaces. The system uses *geometric managers* to take care of managing layouts. By letting geometric managers to do the job has many advantages. For example, your layouts will work on any platform. Also, the more complex interfaces you design, the easier it will be to maintain them using geometric managers. Geometric managers allow you to create truly reusable user interface code.

You will be able to change gadget texts, icon sizes, insert new controls into your windows etc. and the layout is automatically updated for you.

Packer geometry manager allows us to design horizontal and vertical layouts i.e. align controls horizontally or vertically. This is the most often needed geometry manager.

To create a packer:

```
include("oops/r3packer.js");
packer = new r3Packer(R3PA_Orientation, R3PAOF_HORIZONTAL);
```

Then you can ask the window to use the created geometry manager by calling:

```
win.SetGmanager(packer);
```

Anything you insert into this geometry manager gets automatically aligned.

Just like the r3Window class, geometric managers are also derived from the widget base class. However, geometric managers do not have any visible geometric properties.

Creating gadgets

Properties for the button object are defined in the 'oops/r3button.js' header file. You can create a button by calling:

```
include("oops/r3button.js");
```

```
myButton = new r3Button(R3WGA_Parent, myWindow,
                        R3GA_Text, "My Button",
                        R3GA_ToolTip, "Click this button");
```



Note

Just like with windows, you must specify the parent window for the button using the R3WGA_Parent tag. In fact, all user interface objects require that you pass the parent object to the constructor function. The underlying operating system is not able to create the control if no parent is specified.

Below you can find some examples for creating other commonly needed controls.

```
include("oops/r3checkb.js");

gad = new r3Checkbox(R3WGA_Parent, myWindow,
                     R3GA_Text, "A checkbox",
                     R3GCBA_Checked, TRUE);

include("oops/r3cycle.js");

gad = new r3Cycle(R3WGA_Parent, window,
                  R3GA_Text, "Quality",
                  R3MXILTGA_Labels, ["Preview","Medium","Good","Best"],
                  R3GA_ToolTip, "Select desired rendering quality");

include("oops/r3radiob.js");

gad = new r3Radiobutton(R3WGA_Parent, window,
              R3GRBA_Labels, ["Circle","Rectangle","Triangle"]);

include("oops/r3slider.js");

slider =  new r3Slider(R3WGA_Parent, window,
                       R3GA_Text, "Level of detail",
                       R3GSLA_Min, 0,       // minimum level
                       R3GSLA_Max, 100,     // maximum level
                       R3GSLA_Level, 20);   // current level
```

Making objects talk to an application

If we executed the above code, it would create a dummy button object. Clicking the button would do nothing. For all practical applications, we need to make GUI objects to talk to us so that we can link functionality to our user interface. This can be done through *callback functions.*

Let's imagine we want to implement a close button i.e. when the user clicks any of the buttons, the entire application is shut down.

To do this, we define a callback function which calls r3Exit() function:

```
function myhook(window, event, value)
{
    r3Exit();
}
```

Then we pass this function to the button constructor by using the R3RA_Hook tag.

```
button = new r3Button(R3RA_Hook, myhook,
```

```
                        R3WGA_Parent, window,
                        R3GA_Text, "My button");
```

Whenever the user clicks the above button, the 'myhook' function gets called, which shuts down the program.

You can use callbacks with all the basic Realsoft 3D GUI objects, such as slider gadgets, string gadgets, windows, just to name a few.

Putting it all together

Now that we know how to create windows, packers and buttons, let us create a window with horizontally aligned buttons.

```
include("oops/r3button.js");
include("oops/r3window.js");
include("oops/r3packer.js");

function myhook(window, event, value)
{
    r3Exit();
    return 1;
}

window = new r3Window(R3WGA_Parent, _r3gui,
                      R3WA_Title, "My Window",
                      R3WA_ReportCloseWindow, TRUE,
                      R3WA_ReportNewSize, TRUE);

packer = new r3Packer(R3PA_Orientation, R3PAOF_HORIZONTAL);
window.SetGmanager(packer);

for(i = 0; i < 5; i++) {
  button = new r3Button(R3RA_Hook, myhook,
                        R3WGA_Parent, window,
                        R3GA_Text, "Exit " + i);
  packer.ADD(R3PAPF_EXPAND | R3PAPF_FILLX | R3PAPF_FILLY, 0, button);
}

window.REALIZE();
```

This creates the following window:

In the 'for' loop, the program creates five button objects and inserts them into the packer by calling the 'ADD' method. The parameters for this method specify how the packer will manage the control in question.



A window with horizontal packer.

R3PAPF_EXPAND asks the object in question to request as much space as possible from the packer. If there are multiple controls defining expand option, the packer will distribute the space evenly to all requesters. The R3PAPF_FILLX and R3PAPF_FILLY flags then control whether the object should fill the given screen space horizontally, vertically or both.

If you only specify R3PAPF_EXPAND, but you do not specify R3PAPF_FILL options, the buttons will be distributed over the entire window (i.e. the packer will distribute them more space) but the buttons will not be stretched to fill the extra screen space.

The program also calls the FIT method. By passing the R3WFP_BESTFIT parameter to this method, we ask the window to compute the size needed to make all the created sub controls to fit into the window. The size needed depends on the current font, operating system, and many other properties.

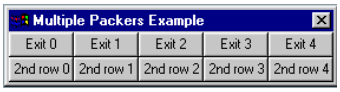
You can find the above sample program from:

scripting/intro/

Creating more complex layouts

Each window can have only one geometric manager associated with it. However, geometric managers can be grouped hierarchically to construct more complex layouts.

For example, we can create a tool bar window consisting of two rows of tools by making the top level packer vertical and then creating two horizontal sub packers for it. The actual tool buttons are then inserted into these sub packers.



A vertical packer two horizontal sub packers.

And here is the code:

```
// create the top level packer
top = new r3Packer(R3PA_Orientation, R3PAOF_VERTICAL);

// create two sub packers
sub1 = new r3Packer(R3PA_Orientation, R3PAOF_HORIZONTAL);
sub2 = new r3Packer(R3PA_Orientation, R3PAOF_HORIZONTAL);

// and insert the sub packers into the top level packer.
top.ADD(R3PAPF_EXPAND | R3PAPF_FILLX | R3PAPF_FILLY, 0, sub1);
top.ADD(R3PAPF_EXPAND | R3PAPF_FILLX | R3PAPF_FILLY, 0, sub2);
```

Then we create and insert desired controls into the sub1 and sub2 packers.

```
// create buttons for the first sub packer
for(i = 0; i < 5; i++) {
    button = new r3Button(R3RA_Hook, myhook,
                          R3WGA_Parent, window,
                          R3GA_Text, "Exit " + i);
    sub1.ADD(R3PAPF_EXPAND | R3PAPF_FILLX | R3PAPF_FILLY, 0, button);
}

// buttons for the second sub packer
for(i = 0; i < 4; i++) {
    button = new r3Button(R3RA_Hook, myhook,
                          R3WGA_Parent, window,
                          R3GA_Text, "2nd row " + i);
    sub1.ADD(R3PAPF_EXPAND | R3PAPF_FILLX | R3PAPF_FILLY, 0, button);
}
```

The geometric manager hierarchy can be as deep as you like, providing you with total control over layouts.

Anchoring

In the previous chapters we demonstrated the usage of pack flags (R3PAPF_EXPAND, R3PAPF_FILLX and R3PAPF_FILLY).

The second parameter for the ADD() method specifies so called 'anchor' flags. Whereas R3PAPF_FILLX and R3PAPF_FILLY flags control the size, the Anchor flags control how the widget is positioned within the space it gets from the packer.

Below you can find the list of available anchor codes.

Code	Description
R3PAAF_CENTER	Align the control at the center
R3PAAF_N	North (up)
R3PAAF_NE	North-East
R3PAAF_E	East (right)
R3PAAF_SE	South-East
R3PAAF_S	South (left)
R3PAAF_SW	South-West
R3PAAF_W	West (down)
R3PAAF_NW	North-West
R3PAAF_ALIGN	Align based on alignment points of controls

Unlike pack options, these are mutually exclusive i.e. you can define only one of them.

Because the anchor codes control how the object in question is managed within the space it gets from the packer, it usually makes sense to use these only with R3PAPF_EXPAND option. If you don't define EXPAND, the inserted control may not have a room to move anywhere.

For example, if you insert a control to a packer by calling:

```
mypacker.ADD(R3PAPF_EXPAND, R3PAAF_CENTER, mybutton);
```

the inserted button will be centered within the space it gets from the geometry manager.

Perhaps one of the most often needed anchor options is R3PAAF_ALIGN, so let's check this out in more detail.

Let's create a window that consists of a number of vertically stacked sliders. The example looks pretty much the same as the first button example we created in the 'Putting it all together' chapter. The only difference is that we now use sliders instead of buttons and use vertical packing rather than horizontal packing.

As usual, we need to include the slider's header file. By studying the header file we can find out the name of the constructor as well as the attributes the slider defines. That's all we need to know in order to create a slider.

Here is the code:

```
include("oops/r3slider.js");
include("oops/r3window.js");
include("oops/r3packer.js");

window = new r3Window(R3WGA_Parent, _r3gui,
                      R3WA_Title, "Align Sliders",
                      R3WA_ReportCloseWindow, TRUE,
                      R3WA_ReportNewSize, TRUE);

packer = new r3Packer(R3PA_Orientation, R3PAOF_VERTICAL);
window.SetGmanager(packer);

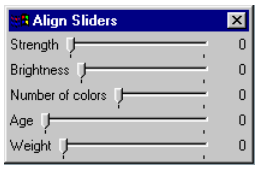
// create five sliders
var sliderLabels = ["Strength", "Brightness", "Number of colors",
                    "Age", "Weight"];

for(i = 0; i < sliderLabels.length; i++) {
    slider = new r3Slider(R3WGA_Parent, cbWindow,
```

```
                            R3GA_Text, sliderLabels[i],
                            R3GSLA_Min, 0,
                            R3GSLA_Max, 100);
        packer.ADD(R3PAPF_EXPAND | R3PAPF_FILLX, 0, slider);
    }

    window.REALIZE();
```

This creates the window shown on the right.



A window with vertically packed sliders.

By inserting the sliders with the R3PAAF_ALIGN, we can align the sliders based on their 'alignment point'.

Let's change the ADD method to:

```
    packer.ADD(R3PAPF_EXPAND | R3PAPF_FILLX, R3PAAF_ALIGN, slider);
```

Run the script. The resulting window is shown on the right.



Sliders inserted with R3PAAF_ALIGN

Creating dynamic interfaces

When designing user interfaces, you should hide unnecessary complexity from the end user. For example, if you select a certain type of geometric object, your tool window canA automatically show only those tools which can be applied to the selected objects.

Geometric managers define the 'Stealth' attribute, which allows you to hide the controls associated with the geometric manager in question. Setting 'Stealth' to 'TRUE' makes all the managed controls invisible and collapses them to zero size. The effect is that the contents of the geometric manager can not be seen and do not require any screen space.

We strongly encourage you to take advantage of this powerful feature so let's go through an example demonstrating this.

Let us create a window which consists of a bunch of sliders. Let's imagine that some of these sliders should be shown only when the user clicks an 'Advanced' check box.

This is really trivial to implement. The only thing we need to do is to put those advanced controls into a separate geometry manager, which we can then either show or hide easily.

Here is the code creating the geometry managers and the sliders:

```
    // create a window and top level packer as usual
    window = ... [snip]
    packer = ...[snip]

    // insert couple of controls into the packer, as usual
    ... [snip]

    // create the 'Advanced' check box.
```

```
checkb = new r3Checkbox(R3RA_Hook, showAdvanced,
                        R3WGA_Parent, window,
                        R3GA_Text, "Advanced");
packer.ADD(R3PAPF_EXPAND | R3PAPF_FILLX | R3PAPF_FILLY, 0, checkb);

// create and insert advanced controls into a separate packer.
packAdvanced = new r3Packer(R3PA_Orientation, R3PAOF_VERTICAL);
packer.ADD(R3PAPF_EXPAND | R3PAPF_FILLX, 0, packAdvanced);

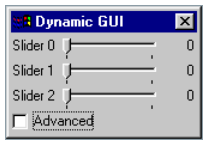
for(i = 0; i < 2; i++) {
    slider = new r3Slider(R3WGA_Parent, window,
                          R3GA_Text, "Advanced " + i);
    packAdvanced.ADD(R3PAPF_EXPAND | R3PAPF_FILLX, 0, slider);
}

// Let's hide the advanced controls by default
packAdvanced.SetStealth(TRUE);

// tell the check box where to find the 'advanced'packer and
// the parent window so that check boxes callback can access them.
checkb.packer = packAdvanced;
checkb.window = window;

// realize, as usual
[snip]
```

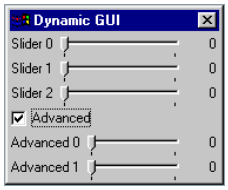
And here is the callback:




```
function showAdvanced(gadget, event, checked)
{
    if(checked)
        gadget.packer.SetStealth(FALSE);
    else
        gadget.packer.SetStealth(TRUE);
    gadget.window.FIT(R3WFP_BESTFIT);
}
```

When the user checks the Advanced check box, the advanced packer is shown. When the user resets the check box, packer is hidden.

Whenever the layout of a window is changed, we need to ask the window to update its layout by calling the FIT method.



Window layout controlled by 'Advanced' check box.

Managing Geometric Objects

Creating Geometric Objects

In the previous tutorial we learned to create various user interface objects, such as windows and buttons. Creating geometric objects is equally easy. For example, in order to create sphere objects, you first need to include the header file defining properties for the sphere class. This can be done by calling:

```
include("real/objects/r3sphere.js");
```



Note

You can find all built-in geometric objects from the 'scripts/js/real/objects' folder. Plug-in libraries may define geometric objects too, in which case their header files can be found from 'scripts/js/plugins' folder.

Once you have loaded the header file, you can create any number of sphere instances. For example, the following program creates two sphere objects.

```
sphere1 = new r3Sphere(0);
sphere1.SetRadius(0.1);

sphere2 = new r3Sphere(0);
sphere2.SetRadius(0.01);
sphere2.SetCenter(new r3Vect(0.1, 0.1, 0));
```

The above code should be self explanatory.

Inserting into the Current Project

Once you have created geometric objects, you also need to decide what to do with them. The most common operation is to insert them into the current project.

To do this, you need to find the JavaScript object managing geometric objects in the current project. This can be done by calling:

```
primLayer = GetJS("CurrentProject.Geometrics");
```



Note

The GetJS() function corresponds to the Get() function. However, the difference is that Get() returns you a Realsoft 3D object, whereas the GetJS() function creates and returns JavaScript wrapper object.

Then you can insert the two spheres into the current project by calling:

```
primLayer.LOCKEXCLUSIVE();
primLayer.INSERT(0, 0, sphere1);
primLayer.INSERT(0, 0, sphere2);
primLayer.RELEASE();
```

When modifying the contents of a layer, you must always lock the layer by calling either LOCKEXCLUSIVE() or LOCKSHARED() methods. The reason for this is that Realsoft 3D is multi threaded application and two or more threads can access the contents of a layer simultaneously.

If you are going to change the contents of a layer, a mutually exclusive 'write' lock must be obtained by calling the LOCKEXCLUSIVE() method. For example, when you insert a new object into a layer, mutually exclusively lock must be used.

In case you only want to read the contents of a layer, shared read lock is sufficient. This can be accomplished by calling the LOCKSHARED() method.

Both type of locks can be released by calling the RELEASE() method.

Creating Levels

To create a level object:

```
include("real/objects/r3level.js");

level = new r3Level(0);
```

Let's imagine we want to create couple of sub objects for the newly created level object, say two cubes.

```
include("real/objects/r3cube.js");

cube1 = new r3Cube(0);
cube1.SetParent(level);
cube1.SetP0(new r3Vect(0.0, 0.0, 0.0);
cube1.SetP1(new r3Vect(0.2, 0.0, 0.0);
cube1.SetP2(new r3Vect(0.0, 0.2, 0.0);
cube1.SetP3(new r3Vect(0.0, 0.0, 0.2);

cube2 = new r3Cube(0);
cube2.SetParent(level);
cube2.SetP0(new r3Vect(0.1, 0.1, 0.0);
cube2.SetP1(new r3Vect(0.25, 0.0, 0.0);
cube2.SetP2(new r3Vect(0.0, 0.25, 0.0);
cube2.SetP3(new r3Vect(0.0, 0.0, 0.25);
```

Note that analytic cubes are defined by four points: one corner point plus three associated 'edge' points. The remaining points for the cube are defined internally, so you don't have to (actually, can't) define them.

A level object also defines the 'Boolean' attribute for boolean operations. Let's turn the level object to a boolean operation, so that the two sub cubes gets boolean operated with each other.

```
level.SetBoolean(R3BOOL_AND);
```

Our hierarchical level object is now ready so we can insert it into the current project by calling:

```
primLayer.LOCKEXCLUSIVE();
primLayer.INSERT(0, 0, level);
primLayer.RELEASE();
```

Transforming Objects

All geometric objects define an attribute called 'Matrix', which defines so called local object space for the object. For example, an object can be moved by constructing the desired transformation matrix and concantenating it to the object's matrix.

The Matrix class defines four methods for setting up transformation matrices: translate(), scale(), rotate() and skew().

Let's imagine we need to translate the selected objects by 0.1 meters on the world 'x' axis. To construct a transformation matrix which translates in world x-axis, call:

```
m = new r3Matrix();
m.translate(0.1, 0, 0);
```

Then you can apply the transformation matrix to the selected objects by calling:

```
primLayer.LOCKEXCLUSIVE();
primLayer.TRANSFORM(m);
```

```
primLayer.RELEASE();
```

You may concatenate any number of matrix operations into a single matrix. In this case, the last concatenated operation is executed first. For example, if you want to move and scale the selected objects, call:

```
m = new r3Matrix();
m.scale(0.5 0.5, 0.5);
m.translate(0.1, 0, 0);
```

in which case the selected objects are first translated and then scaled.

Any transformation effect, such as scaling, rotation or skew, or any combination of them can be constructed and applied this way.

Single Point Editing

The TRANSFORM() method allows you to apply any linear transformation to an object. However, it does not allow you to deform the geometry of an object.

The geometry base class ('scripts/js/real/objects/r3prim.js) defines the SETPOINT() method, allowing you to set any of the geometric attributes defined by the object in question.

Let's create a nurbs curve consisting of four control points.

```
include("real/objects/r3nurbs.js");

nurbs = new r3Nurbs(0);
nurbs.SetOrder(4);
nurbs.SetCount(4);
nurbs.SetKnots([0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0]);
for(i = 0; i < 4; i++)
    nurbs.SETPOINT(i, new r3Vect(i * 0.1, 0, 0));
```

Here we used SETPOINT() method to define initial geometry for the nurbs curve.

Another useful method is GETPOINT(), which allows you to fetch the value of any desired geometry point. By using GETPOINT and SETPOINT methods, you can apply deformation effects to geometric objects. First you call GETPOINT() to fetch the current value, then you modify the value using some non linear function and finally you assign the modified value back to the object by calling SETPOINT().

For example, the following uses fractal noise to deform all geometric points of the given object.

```
function applyNoise(obj)
{
    // create new vector object
    p = new r3Vect();

    // fetch the number of geometry points in the given object
    pcount = obj.GetPointCount();

    for(i = 0; i < pcount; i++) {
        // fetch the current value of the point 'i'
        obj.GETPOINT(p, i);

        // displace the point using fractal noise
        p2 = p.noise(3, 4);
        p2.fmul(0.1);
```

```
            p.fadd(p2);

            // set displaced point back to the object
            obj.SETPOINT(i, p);
        }
    }
```

Enumerating Selected Objects

Now that we managed to implement a somewhat interesting deformation effect, let's see how it could be plugged in as a new deformation tool.

The layer base class defines a method which is very useful when implementing custom tools. The method is called ENUMSELECTLIST(). This method scans through the currently selected objects and passes them to the given hook function, such as the deformation function we wrote above.

There is only one complication. Because the JavaScript objects are 'wrapper' objects attached to actual Realsoft 3D objects, our hook function will be called with a Realsoft 3D object rather than a JavaScript object. The low level Realsoft 3D functionality does not know if the enumeration request was generated by JavaScript wrapper object. Therefore, we must create a JavaScript interface to the given Realsoft 3D object before we can access its properties through JavaScript methods.

Here is the code:

```
function deformByNoiseCallBack(r3obj)
{
    // create JavaScript interface to 'r3obj'
    jsobj = R3ToJS(r3obj);

    // call the applyNoise() function
    if(obj)
        return applyNoise(jsobj);
    return 1;
}
```

Now you can deform the selected objects by calling:

```
primLayer.LOCKEXCLUSIVE();
primLayer.ENUMSELECTLIST([R3RA_Hook, mycallback]);
primLayer.RELEASE();
```

The 'R3RA_Hook' parameter is used for specifying the callback function to be called.

You can find this example from the 'scripts/js/myclasses/toolbars/mydeform.js'. To test it, create couple of SDS or NURBS objects, select them and apply 'Scripts/Toolbars/mydeform.js' pull-down menu.

Vectors and Matrices

Vectors

In the Managing Geometric Objects tutorial we used vector objects in various ways. Most geometric objects define vector attributes and in order to manage geometric objects, you need to know something about vectors.

Vector is one of the built-in classes in Realsoft 3D, so you don't have to include a header file to get the class defined.

A vector object defines three attributes: x, y and z. It can be used for representing geometry points, direction vectors, and many other things of 3D nature.

You can create a vector by calling:

```
v = new r3Vect();
```

which creates so called null vector. All the three attributes are initialized to zero. You can initialize a vector to a desired value by calling the set() method.

```
v.set(1, 0, 0);
```

or, simply by setting the attributes using an assignment operator:

```
v.x = 1.0;
v.y = 0;
v.z = 0;
```

You may also create a vector by defining initial x, y and z values to the constructor:

```
v = new r3Vect(1, 0, 0);
```

Vector class also defines methods for vector addition and other vector operations. For example, let's imagine we have two vectors:

```
a = new r3Vect(1, 0, 0);
b = new r3Vect(0, 1, 0);
```

Then we want to perform a vector operation c = a + b. This can be accomplished as follows:

```
c = a.add(b);
```

This corresponds to the following code:

```
c.x = a.x + b.x;
c.y = a.y + b.y;
c.z = a.z + b.z;
```

Let's take another commonly needed operation like 'scalar multiplication'. For example, if you want to perform operation 'b = a * 2.0' i.e. to double the length of the vector, call:

```
b = a.mul(2.0);
```

This corresponds to the following code:

```
b.x = a.x * 2.0;
b.y = a.y * 2.0;
b.z = a.z * 2.0;
```

There are two methods for applying most vector operations: those which result in a new vector object and those which apply the operation to the vector in question. We demonstrated the former type of methods above.

If a method name starts with letter 'f', then it represents the latter form of method i.e. it applies the vector operation to the object in question rather than resulting in a new vector. If you are familiar with Realsoft 3D Software Development Kit you will find that the method names correspond to those used in the SDK.

For example, the following method:

```
a.fadd(b);
```

adds the vector 'b' to 'a'. This corresponds the code:

```
a.x = a.x + b.x;
a.y = a.y + b.y;
a.z = a.z + b.z;
```

Or, if you want to double the length of a vector, call:

```
a.fmul(2.0);
```

which corresponds to the following code:

```
a.x = a.x * 2.0;
a.y = a.y * 2.0;
a.z = a.z * 2.0;
```

One very useful vector method is called 'noise', which allows you to compute various fractal effects. The noise takes two parameters: dimensions and octaves and results in a new vector object representing the strength of the noise field.

```
b = a.noise(3, 4);
```

For a list of all methods supported by the vector class, see the reference manual/vector class.

Matrices

Realsoft 3D uses a very standard 4x4 matrix to define so called 'local object space' property for all geometric objects. The matrix consists of 16 floating point values (4x4).

The mathematical theory behind matrices is beyond this tutorial. However, it is possible to use the matrix class even if you do not understand the deep theory.

The matrix class is a built-in class and there are no header files that you should include.

Constructing Matrices

As demonstrated in the Managing Geometric Objects tutorial, a new matrix object can be created by calling:

```
m = new r3Matrix();
```

This creates a so called 'identity' matrix. Applying such a matrix to does not transform objects in any way.

You may also pass four vector parameters to the constructor:

```
origin = new r3Vect(0, 0, 0);
x = new r3Vect(1, 0, 0);
y = new r3Vect(0, 1, 0);
z = new r3Vect(0, 0, 1);
```

```
m = new r3Matrix(origin, x, y, z);
```

The parameters define a coordinate system (i.e. a space), which is an alternative (more intuitive) representation for a matrix. For example, to construct a transformation matrix which moves the object from world origin to x=1.0, y=0.5:

```
origin = new r3Vect(1.0, 0.5, 0);
x = new r3Vect(1, 0, 0);
y = new r3Vect(0, 1, 0);
z = new r3Vect(0, 0, 1);
m = new r3Matrix(origin, x, y, z);
```

Or, to construct a matrix which scales the object to twice the original size:

```
origin = new r3Vect(0, 0, 0);
x = new r3Vect(2, 0, 0);
y = new r3Vect(0, 2, 0);
z = new r3Vect(0, 0, 2);
m = new r3Matrix(origin, x, y, z);
```

A third way to construct a matrix is by passing sixteen number parameters to the constructor, specifying the initial state of the sixteen attributes of the matrix. However, this is not very convenient and rarely used.

Constructing Transformation Matrices

Many applications allow you to construct matrices from a number of transformation components namely translate, scale, rotate and skew.

Concatenated transformations are executed in the reversed order (first in, last out).

Let's demonstrate this by going through a couple of examples.

If you run the following code:

```
m = new r3Matrix();
m.scale(0.5, 0.5, 0.5);
primLayer.LOCKEXCLUSIVE();
primLayer.TRANSFORM(m);
primLayer.RELEASE();
```

the result is that all the selected objects shrink towards the world origin so that they end up twice as small.

However, let us imagine you want to scale the selected objects about point [0.1, 0.2, 0.0] rather than about the origin.

To do this, you first have to move the object, so that this point matches the world origin. Then you apply the scale effect and finally you move the object back to its original position. The code to accomplish this is:

```
m = new r3Matrix();
m.translate(0.1, 0.2, 0.0);
m.scale(0.5, 0.5, 0.5);
m.translate(-0.1, -0.2, 0.0);

primLayer.LOCKEXCLUSIVE();
primLayer.TRANSFORM(m);
primLayer.RELEASE();
```

Because specified transformations are executed in reversed order, the translation [-0.1, -0.2, 0] is executed first. Then the object is scaled and finally moved back to its original position. The effect is that the object seems to be scaled down towards the point [0.1, 0.2, 0].

Specifying transformation center in one call

The transformation methods accept more than three parameters, allowing you to specify transformation center in one call.

The above scaling effect can easily be set up, simply by calling:

```
m = new r3Matrix();
m.scale(new r3Vect(0.1, 0.2, 0), 0.5, 0.5, 0.5);
```

SIn other words, if you pass four parameters to the scale method, the first parameter is assumed to be the origin of scale. The remaining three parameters specify scaling factors in x, y and z dimensions.

Let's take another example: to construct a transformation matrix rotating the selected objects about an axis aligned to world 'x' axis and passing through the point [0, 1, 0], call:

```
m = new r3Matrix();
m.rotate(PI, new r3Vect(0, 1, 0), new r3Vect(1.0, 0.0, 0.0));
```

The first parameter specifies rotation angle in radians. The second parameter specifies the origin of rotation and the third specifies the axis of rotation.

The script 'scripts/js/myclasses/mytransform.js' shows you couple of examples of tools transforming geometric objects. You can test these tools by selecting the pull down menu 'Scripts/Toolbars/myransform.js'.

Managing Materials

Creating a Material

In this tutorial we show how one can manage VSL (Visual Shading Language) material objects using JavaScript. Basic understanding of VSL is assumed.

VSL Material and other VSL specific objects are defined in the r3code library. Correspondingly, the header files for them can be found from 'scripts/js/real/code' folder.

To create a VSL material:

```
include("real/code/r3vsl.js");

mat = new r3Vsl(0);
mat.SetName("my material");
```

The above creates an empty VSL material object named 'my material'.

In order to let a VSL material define surface properties, you need to create and insert a so called surface shader object into the VSL material.

```
include("real/code/r3mpphas.js");   // shader

shader = new r3Mpphaselevel(0);
shader.SetPhase(R3CLID_PHASESURFACE);
```

```
vsl.INSERTPROP(0, 0, shader);
```

The above code creates a 'phase level' VSL object and sets its type to R3CLID_PHASESURFACE. VSL objects inside this level will be able to access and define various surface properties, such as surface color.

Let's create a texture mapping object to define the surface color property:

```
include("real/code/r3mptext.js");   // texture map

texture = new r3Mptexture(0);
texture.SetParent(shader);
texture.SetImageName("textures/lake.jpg");
```

By calling the SetParent() method you link the texture object to the shader object. The SetImageName() method asks the texture mapping object to use the "textures/lake.jpg" image.

To insert the material into the current project, you need to fetch the JavaScript class managing material objects in the current project.

```
matLayer = GetJS("CurrentProject.Materials");
```

Then you can insert the material into the current project by calling:

```
matLayer.LOCKEXCLUSIVE();
matLayer.INSERT(0, 0, vsl);
matLayer.RELEASE();
```

Reflection Mapping

The technique where an image is used for defining the surface color is called texture mapping. In Realsoft 3D, images can be used for defining any material property, such as Brilliancy, Transparency, Refraction Factor, Specularity, just to name a few.

The reason why the previous example resulted in texture mapping rather than some other type of mapping is that the default output channel for the texture mapping object is 'Color'.

There are many other channels available. Each channel is identified by two properties: class and name. For example, renderer uses the channel object whose class is R3CLID_COLORCHANNEL and name is 'Color' to represent surface color.

So, to get 'reflection mapping' rather than 'color mapping', we just need to assign the texture mapping object to the 'Reflection' channel:

```
texture.SetOutputChannel(R3CLID_REFLECTIONCHANNEL);
texture.SetOutputName("Reflection");
```

This will turn the texture mapping into reflection mapping.

Creating VSL Curve Objects

The VSL Curve object (r3Mpcurve) is defined in the 'scripts/js/real/code/r3mpcurv.js' file. However, what makes it interesting is that it does not define any curves itself. Instead, it uses the r3Curve object for this. So, in order to create a curve object we need to know how to fetch the actual curve object and how to access the curve data in it.

Let's create a material with a curve object:

```
include("real/code/r3mpcurv.js");

// create a VSL material
vsl = new r3Vsl(0);
vsl.SetName("curve");

// create a surface shader object for the material
shader = new r3Mpphaselevel(0);
shader.SetPhase(R3CLID_PHASESURFACE);
vsl.INSERTPROP(0, 0, shader);

// create a VSL curve object into the shader object
mpcurve = new r3Mpcurve(0);
mpcurve.SetParent(shader);

// attach to color channel
mpcurve.SetOutputChannel(R3CLID_COLORCHANNEL);
mpcurve.SetOutputName("Color");
```

So, we have now created a VSL curve object and attached it to the Surface Color channel.

In order to add some points into the curve object, we must first fetch the actual curve object from the VSL curve object. This can be done by calling the GetCurve() method.

```
curve = mpcurve.GetCurve();
```

The above call returns r3Curve object (scripts/js/real/code/r3curve.js).

By default all curve objects have two points, defining a linear curve. Let's get rid of these default curves. To delete existing points from the curve, call:

```
curve.DELETEALLPOINTS();
```

Then we can add a number of points to the curve by calling ADDPOINTIFNOTONLINE method. Let's add three points to the curve:

```
curve.ADDPOINTIFNOTONLINE(R3TID_VECTOR, new r3Vect(0, 0, 0),
    new r3Vect(0.3, 0.2, 0.1));
curve.ADDPOINTIFNOTONLINE(R3TID_VECTOR, new r3Vect(0.5, 0.5, 0.5),
    new r3Vect(0.4, 0.1, 0.2));
curve.ADDPOINTIFNOTONLINE(R3TID_VECTOR, new r3Vect(1, 1, 1),
    new r3Vect(0.9, 0.7, 0.6));
```

The first vector value specifies x coordinates for the point to be added. The second vector parameter specifies corresponding y values.

Because we attached the curve object to Color channel, the curve defines three sub curves: one for each color sub channel (red, green, blue). Therefore, we must pass vectors to the ADDPOINTIFNOTONLINE method. If we attached the curve object to say the Alpha channel, there would be only one curve. Correspondingly, we would call:

```
curve.ADDPOINTIFNOTONLINE(R3TID_FLOAT, 0.1, 0.5);
```

The final step is to insert the VSL material into the current project:

```
matLayer.LOCKEXCLUSIVE();
```

```
matLayer.INSERT(0, 0, vsl);
matLayer.RELEASE();
```

Layers

In the Select Window tabs, the user can manage various type of objects, such as geometric primitives, materials, rendering settings and so on. Each of the Select Window tabs correspond to a certain layer object. For example, the material tab shows you the contents of the 'material layer' of the current project. The Grid tab shows you the contents of the 'grid layer' of the current project.

We have already explored some of these layer objects, such as geometric primitives and material objects.

All layer objects have a common base class, called 'scripts/js/real/layer/r3objlay.js'. This class defines methods such as INSERT(), LOCKEXCLUSIVE() and RELEASE().

Let's go through a couple of other useful methods defined by the object layer class.

Exporting Selected Objects

The layer object defines a property called 'selected objects'. Many of the tools in Realsoft 3D tools are actually applied to selected objects. For example, the Move tool translates selected geometric objects. We have already used the method ENUMSELECTLIST to deform objects. Let's demonstrate another enumeration method called ENUMOBJECTS. This method enumerates the entire contents of a layer.

Let's assume we want to export geometric objects to a file named 'export.js'.

To create a Realsoft 3D file object, call:

```
myFile = new r3File(R3FIA_FileName, "export.js", R3FIA_Mode, "wa");
```

Then we need a hook function for handling geometric objects. In the hook function, we create a JavaScript wrapper object for the enumerated Realsoft 3D geometry object, fetch the attributes of it and write them out to the opened file. For sake of simplicity, we handle only the sphere object here. It would be a very straight forward task to add support for other geometric objects as well.

```
include("real/objects/r3sphere.js");

function myExportHook(r3obj)
{
  obj = R3ToJS(r3obj);
  if(!jso)
      return 1;

  // fetch the class identifier of the object
  iClid = obj.GetClassID();

  if(clid == R3CLID_SPHERE) {

    sName = obj.GetName();
    pCenter = obj.GetCenter();
    vA = obj.GetA();
    vB = obj.GetB();
    vC = obj.GetC();

    // write them out
```

```
      this.myFile.PUTS("ClassID: " + iClid + "(ellipsoid)\n");
      this.myFile.PUTS("Name: " + sName + "\n");
      this.myFile.PUTS("Center: {" + pCenter.x + ", " + pCenter.y +
                       ", " + pCenter.z + "}\n");
      this.myFile.PUTS("A: {" + vA.x +  ", " + vA.y + ", " + vA.z + "}\n");
      this.myFile.PUTS("B: {" + vB.x + ", " + vB.y + ", " + vB.z + "}\n");
      this.myFile.PUTS("C: {" + vC.x + ", " + vC.y + ", " + vC.z + "}\n");
    }
    return 1;
  }
```

Then you can export objects by calling the ENUMOBJECTS method:

```
primLayer.LOCKSHARED();
primLayer.ENUMOBJECTS([R3RA_Hook, myExportHook]);
primLayer.RELEASE();
```

Setting Attributes to the Selected Objects

Another often needed task is to set desired attributes to the selected objects. There are many ways to do this. For example, you can use the ENUMSELECTLIST method to enumerate selected objects to a hook function, which then checks the type of the object and sets appropriate attributes.

Another approach is to call the SETONSELECTED method. This methods accepts a tag list, specifying attribute identifiers, and corresponding values. For example, let's assume we need to set spin and velocity for the selected objects. Velocity and Spin are attributes defined by 'scripts/js/real/objects/r3prim.js' class, which is the base class for all geometric objects.

```
vel = new r3Vect(0.1, 0.1, 0);
spin = new r3Vect(PI * 2, 0, 0);

primLayer.LOCKEXCLUSIVE();
primLayer.SETONSELECTED([R3PRIMA_Mass, 1.2,
                         R3PRIMA_Velocity, vel,
                         R3PRIMA_Spin, spin]);
primLayer.RELEASE();
```

Implementing New Classes

JavaScript supports a basic object oriented paradigm, allowing you to implement new kinds of objects by deriving them from existing objects.

In this tutorial we show you how to implement new JavaScript classes in Realsoft 3D. We also explore the internal structure of Realsoft 3D and demonstrate how to transform geometric objects, create new geometric objects, export objects to a file and create new material objects.



Note

This tutorial is not a JavaScript tutorial. Basic knowledge of JavaScript's object oriented programming model is assumed.

Class Header Files

In the previous tutorials we used the include() function to load various header files.

For example, we loaded button's header file:

```
include("oops/r3button.js");
```

Each of these header files define properties for one Realsoft 3D class. For example, 'scripts/js/oops/r3button.js' defines the JavaScript class corresponding to the Realsoft 3D Button class.

The substructure of 'scripts\js' folder corresponds to the library structure of Realsoft 3D. For example, the 'oops' sub folder stands for Realsoft's Object Oriented Programming System and it defines classes for the r3oops library. If you buy a plug-in that supports JavaScript you get a new folder that contains the header files corresponding to the classes in the plug-in library.

You can also add new sub folders into 'scripts/js' folder for your own JavaScript classes. In fact, the scripts folder should already have 'myclasses' sub folder, which contains a number of sample classes.

myWindow Class

Any of the sample scripts we have gone through so far can be turned to JavaScript classes easily.

Let's recall the first sample we went through in the previous Intro tutorial:

```
include("oops/r3window.js");

window = new r3Window(R3WGA_Parent, _r3gui,
                      R3WGA_Left, 300,
                      R3WGA_Top, 200,
                      R3WGA_Width, 300,
                      R3WGA_Height, 200,
                      R3WA_ReportCloseWindow, TRUE,
                      R3WA_ReportNewSize, TRUE,
                      R3WA_Title, "Dummy Window");

window.REALIZE();
```

Creating window objects is straightforward, but specifying all the above attributes creates a lot of typing overhead.

Let's assume we want to create a new JavaScript window, which creates the above kind of a window with minimal typing overhead. This can be done by deriving a new class which takes care of setting the appropriate attributes for us.

```
include("oops/r3window.js");

// constructor function
function myWindow(name, width, height)
{
    if(!arguments.length)
        return;
    this.base = r3Window;

    // our super class
    this.base(R3WGA_Parent, _r3gui,
              R3WGA_Left, 300,
              R3WGA_Top, 200,
              R3WGA_Width, width,
              R3WGA_Height, height,
              R3WA_ReportCloseWindow, TRUE,
```

```
                R3WA_ReportNewSize, TRUE,
                R3WA_Title, name);
    }
    // derive 'myWindow' from the 'r3Window' base class
    myWindow.prototype=new r3Window;
```

This is how to derive new classes in JavaScript. If you save the above code as 'js/myclasses/mywindow.js', you will be able to create windows as follows:

```
    include("myclasses/mywindow.js");

    win = new myWindow("My cool window", 300, 200);
    win.REALIZE();
```



Note

You should not put the R3WGM_REALIZE call into a constructor. This method should be called only once from the highest level (whoever creates your object is responsible also for realizing them). If you put the realize method into the constructor function of your class, it gets called multiple times causing flickering. Another method which you should not put into constructor functions is 'R3WGM_FIT'.

Implementing 'myToolBar' Class

Now that we know how to derive new JavaScript classes for existing classes, let's go through a more practical example. Let's create a fully functional tool bar class, which allows us to create custom tool bars as easily as possible.

Recall the packer example we went through in the beginning of the intro tutorial. We created a window with a packer geometric manager and then inserted a bunch of buttons into the packer. Typically toolbars contain one or more horizontally or vertically packed tool buttons. So we can take the packer example as a starting point for developing our tool bar class.

Again, we need to wrap the creation code into a constructor function to get a proper JavaScript class defined. Because the tool bar should be user configurable, we just create a window with an empty packer in the constructor. Then we implement another method which allows us to insert tool buttons into the toolbar.

So, let's see how all this can be implemented.

Here is the code:

```
    // our tool bar consists of a window, a packer and one or more buttons
    include("oops/r3button.js");
    include("oops/r3window.js");
    include("oops/r3packer.js");

    // let the user to control the orientation of the tool bar
    var MYTB_HORIZONTAL = 0
    var MYTB_VERTICAL = 1

    // add button with user specified label and callback function
    function mytbAddTool(buttonLabel, fnCallBack)
    {
        button = new r3Button(R3RA_Hook, fnCallBack,
                              R3WGA_Parent, this,
                              R3GA_Text, text);
        if(button) {
            // insert button into the packer
```

```
        this.packer.ADD(R3PAPF_FILLX, 0, button);

        // tell the button where to find the project
        button.layer = this.layer;
    }
    return button;
}

// constructor: create a window with empty packer
function myToolBar(titlebar, orientation, layer)
{
    if(!arguments.length)
        return;
    this.base = r3Window;
    this.layer = layer;

    this.base(R3WGA_Left, 200,
              R3WGA_Top, 150,
              R3WA_ReportNewSize, TRUE,
              R3WA_ReportCloseWindow, TRUE,
              R3WA_Title, titlebar);

    // create a packer with given orientation
    this.packer = new r3Packer(0);
    this.packer.SetOrientation(orientation);

    // define method for inserting tool buttons
    this.AddTool = mytbAddTool;

    // and give the packer to the window, as usual
    this.SetGmanager(this.packer);
}

myToolBar.prototype=new r3Window;
```

Testing 'myToolBar' Class

The above JavaScript program can be found from 'scripts/js/myclasses/toolbars/mytoolbar.js'.

Let's add a tool button which creates an analytic sphere object. For this we need to create a call back function. This function is called when the user clicks the tool button. In the function we simply create a sphere and insert it into the geometry layer.

```
function mySphereCallback(button, event, value)
{
    // create a sphere
    sphere = new r3Sphere(0);

    // set some sphere specific attributes
    sphere.SetRadius(0.1);
    sphere.SetCenter(new r3Vect(0.1, 0.0, 0.0));
    sphere.SetName("my sphere");

    // insert the sphere into the project
    button.layer.LOCKMUTEX(0, 0, sphere);
    button.layer.INSERT(0, 0, sphere);
    button.layer.RELEASE();
}
```

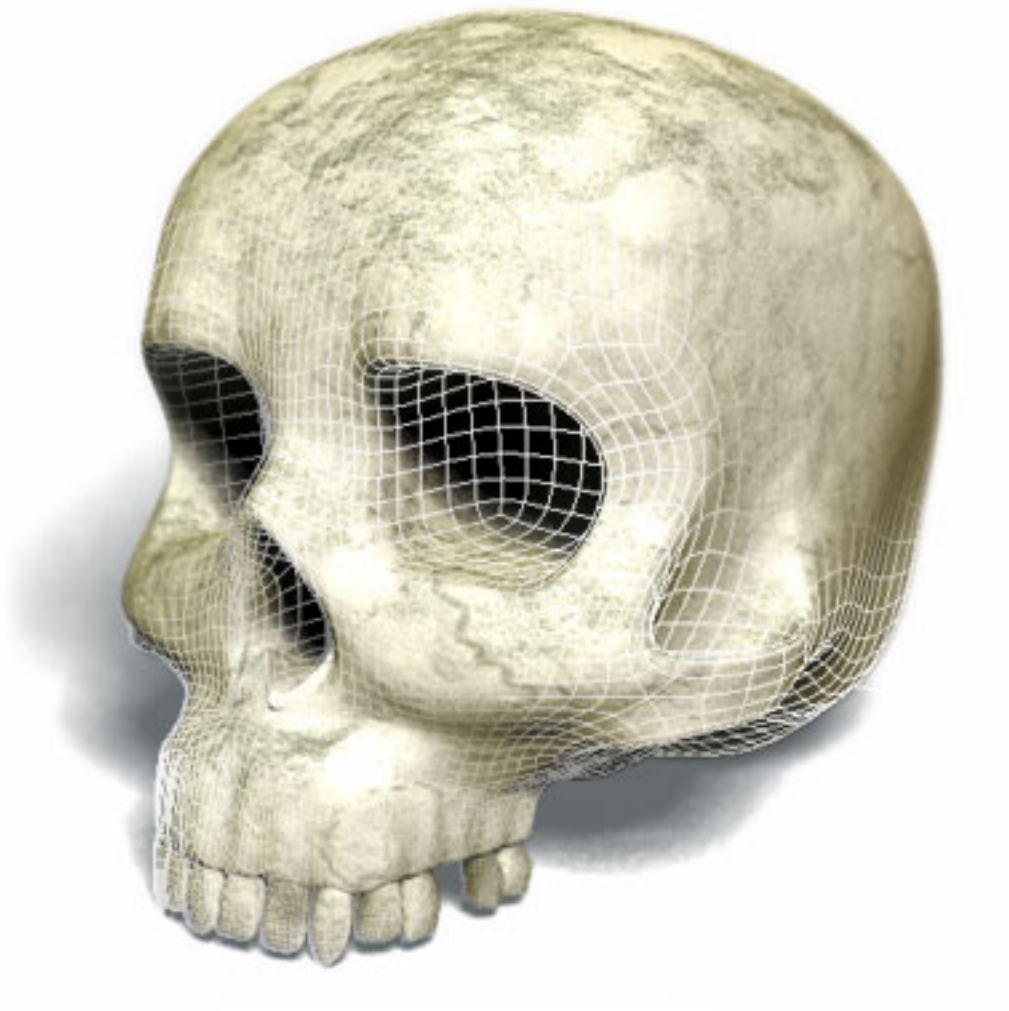
Now we have defined one callback which can create a sphere into the current layer. We have also defined a new tool bar class.

To actually create a tool bar, enter:

```
tbar = new myToolBar(MYTB_HORIZONTAL, primLayer);
tbar.AddTool("Sphere", mySphereCallback);
```

When you click the 'Sphere' button, the 'mySphereCallback' is called and new sphere will appear into the current project.

The 'scripts/js/myclasses/toolbars' folder contains several JavaScript examples derived from the 'myToolBar' class.



Chapter 14. Miscellaneous

Hot Keys

The shortcut keys activate or perform various frequently used program functions.

```
a   Again (repeat latest action)
b   Bend
c   Reserved
d   Drop selected objects to a level
e   Reserved (coordinate lock)
f   Fit view to selected objects
g   Grid on/off
h   OpenGL on/off
i   Invisible on/off
j   View to object
k   Object view
l   Extend (change Length)
m   Move
n   Enter numeric mode
```

```
o   Mirror (Opposite)
p   Open property window
q   Reserved (coordinate lock)
r   Rotate
s   Scale
t   Configure view for modeling in object space
u   Reset from modeling in object space
v   Reserved for view compass
w   Reserved (coordinate lock)
x   Reserved (coordinate lock)
y   Reserved (coordinate lock)
z   Reserved (coordinate lock)

A   Animate by path / Add SDS edge

B   Create Backdrop / SDS bevel
C   Create Camera from the current view
D   Create Subdivision cube / Destroy SDS face
E   Reserved for modifiers
F   Font tool
G   Create group
H   Extrude tool (H for Height) / Extrude SDS face
I   Create vertical axis line (looks like I)
J   Create NURBS curve by control points
K   Create NURBS curve by knotpoins
L   Create Level
M   Create MetaBall / Merge SDS vertices

N   Create NURBS mesh
O   Create analytic cylinder (O-like cross section)
P   Particle tool
Q   Reserved for modifiers
R   Create analytic Rectangle
S   Create analytic Sphere / Subdivide SDS face / Split SDS edge
T   Parallel (Texture) mapping tool / Tunnel SDS faces
U   Create analytic cube
V   Create analytic cone (sharp V like shape)
W   Reserved for modifiers
X   Reserved for modifiers
Y   Reserved for modifiers
Z   Reserved for modifiers

Ctrl a  Select all
Ctrl b  Define render box for view window
Ctrl c  Copy
Ctrl d  Duplicate
Ctrl g  Execute current macro
Ctrl i  Invert selection
Ctrl l  Load file
Ctrl m  Macro recording on/off
Ctrl n  New project
Ctrl p  Pause/continue macro recording
Ctrl q  Save active view window contents to an image file
Ctrl s  Save file
Ctrl t  Deselect
Ctrl r  File render window
Ctrl v  Paste
Ctrl x  Cut
Ctrl y  Redo
```

```
Ctrl z  Undo


1   Reserved as modifier (iplane horizontal lock)
2   Reserved as modifier (iplane vertical lock)
3   Reserved as modifier (iplane normal lock)
4   Reserved as modifier (lasso selector)
5   Reserved as modifier (snap to nearest point)
6   Reserved as modifier (snap to curve)
7   Front View
8   Side View
9   Top View
0   Ray trace current view


Space     Edit mode on/off
Alt Space Selection Painter
Home      Reset view
Insert    Move pivot point
Del       Delete selected object/point/face etc.

!   Animation record on/off
§   Create distant light
@   Create ambient light
£   Create point light
$   Create spot light
&   Boolean And
%   Boolean And Not
/   Hide wire geometry
*   Perspective on/off

-   Backwards 1 frame
+   Forward one frame
>   Play animation/stop animation
<   Rewind animation
)   Tracking on & play animation
(   Camera tracking off
#   Ground plane drawing on/off
;   Reset grid origin
=   Reserved

?   Unused
[   Unused
]   Unused
{   Unused
}   Unused

F1   Reserved
F2   Select window show/hide
F3   Toolbar show/hide
F4   Control bar show/hide
F5   View bar show/hide
F6   Choreographs
F7   Animation bar show/hide
F8   Numeric show/hide
F9   Status show/hide
F10  Reserved
F11  UV editor
F12  Color window
```

```
Ctrl F2  Info window
Ctrl F3  Scripting window
Ctrl F4  View property window
Ctrl F5  Floating view window
```

Modifier keys

Modifier keys affect the performed program functions while they are pressed down. They do not start any operation or tool.

Input plane constraints

```
1               Lock to horizontal direction of input plane
2               Lock to vertical direction
3               Lock to normal direction
Shift           Lock to nearest axis
```

Object space constraints

```
q               Lock to horizontal direction of object space
w               Lock to vertical direction
e               Lock to normal direction
```

World space constraints

```
x               Lock to horizontal direction of absolute space
y               Lock to vertical direction
z               Lock to normal direction
```

Grid control by mouse click/dragging

```
. (dot)         Set hotpoint
Alt . (Alt+dot) Set hotpoint for all views
```

Snapping

```
5               Snap to nearest point
6               Snap to curve
Ctrl            Snap to selected objects only
Alt             Snap to current level objects only
Alt             Snap to existing points in SDS object face creation
Shift           Grid snapping off for SDS face, edge and vertex handle moving
                and object translation handles
```

Mouse projection to 3D

```
, (comma)       Project LMB click to underlying surface
, (comma)       3D snap (don't project a dragged point to input plane)
```

Object and handle selection

Shift	Add to selection (OR)
Ctrl	Intersection with previous selection (AND)
Ctrl	Range select: select all from previous selected to clicked one
Shift Ctrl	Invert selection (XOR)
Alt	Subtract from selection
4	Lasso selector in dragging

View navigation by the right mouse button

Shift	Zoom (change view angle/zoom scale)
Ctrl	Pan (move camera)
Alt	Rotate camera about aimpoint
Shift Ctrl	Move camera forward/backward
Shift Alt	Rotate in 45 degree steps
Ctrl Alt	Adjust camera banking
Shift Ctrl Alt	Adjust camera banking in 45 degree steps

View navigation by the middle mouse button

Shift	Move camera forward/backward
Ctrl	Pan (move camera)
Alt	Rotate camera about viewpoint
Shift Alt	Rotate camera about viewpoint in 45 degree steps

View cruising with cursor keys

Tab	View camera cruising on
Tab+Shift	Turbo boosted cruising
Tab+Up	Accelerate
Tab+Down	Decelerate
Tab+Left	Turn left
Tab+Right	Turn right
Tab+Alt+Up	Turn up
Tab+Alt+Down	Turn down
Tab+Alt+Left	Bank counterclockwise
Tab+Alt+Right	Bank clockwise
Tab+Ctrl+Up	Move up
Tab+Ctrl+Down	Move down
Tab+Ctrl+Left	Move left
Tab+Ctrl+Right	Move right
Tab+Left+Right	Turn towards horizontal flight

Create/modify tools

Shift	Uniform 3D scale/45 degree rotate grid/Uniform rectangle and cube
Ctrl	Duplicate at modify (transformation handles only)
Alt	Instance at modify(via handles)/Snap to existing SDS points

Compass menus

```
v compass       View tools
y compass       Light sources & texture tools
q compass       Create tools
w compass       Wire compass
e compass       Modify tools

x compass       Reserved
z compass       Reserved

1 compass       Unused (reserved for user compass)
2 compass       Unused
3 compass       Unused
```

Geometry modify tools

```
b   Bend
i   Invisible on/off
l   Extend (change Length)
m   Move
o   Mirror (Opposite)
r   Rotate
s   Scale
/   Hide/show wire geometry
```

Object hierarchy editing tools

```
d   Drop selected objects to a level
L   Create Level
&   Boolean And
%   Boolean And Not
Ctrl a  Select all
Ctrl c  Copy
Ctrl d  Duplicate
Ctrl i  Invert selection
Ctrl t  Deselect
Ctrl v  Paste
Ctrl x  Cut
```

Animation tools

```
!   Animation record on/off
-   Backwards one frame
+   Forward one frame
>   Play animation/stop animation
<   Rewind animation
)   Tracking on & play animation
(   Camera tracking off
A   Create path animation
```

Light sources

```
§   Create distant light
@   Create ambient light
£   Create point light
```

```
$   Create spot light
```

Gui elements

```
p    Property window
F2   Select window show/hide
F3   Toolbar show/hide
F4   Control bar show/hide
F5   View bar show/hide
F6   Choreographs
F7   Animation bar show/hide
F8   Numeric show/hide
F9   Status show/hide
F11  UV editor
F12  Color window

Ctrl F2  Info window
Ctrl F3  Scripting window
Ctrl F4  View property window
Ctrl F5  Floating view window
Ctrl+r   File render window
Ctrl+l   File load window
```

View window

```
f   Fit view to selected objects
g   Grid on/off
h   OpenGL on/off
j   View to object
k   Object view
t   Configure view for modeling in object space
u   Reset from modeling in object space
*   Perspective on/off
#   Ground plane drawing on/off
;   Reset grid origin
0   Zero key ray traces the current view
7   Front View
8   Side View
9   Top View

Ctrl b  Define render box for view window
Ctrl q  Save view to image file
```

SDS editing

```
A   Add edge
B   Bevel
D   Destroy face
H   Extrude face
M   Merge SDS vertices
S   Subdivide face / Split edge
T   Tunnel faces
```

Creation and construction tools

```
B   Create Backdrop
C   Create Camera from the current view
D   Create Subdivision cube
F   Font tool
G   Create group
H   Extrude tool
I   Create vertical axis line (looks like I)
J   Create NURBS curve by control points
K   Create NURBS curve by knotpoins
L   Create Level
M   Create MetaBall
N   Create NURBS mesh
O   Create analytic cylinder (O-like cross section)
P   Particle tool
R   Create analytic Rectangle
S   Create analytic Sphere
T   Parallel (Texture) mapping tool
U   Create analytic cube
V   Create analytic cone (sharp V like shape)
§   Create distant light
@   Create ambient light
£   Create point light
$   Create spot light
```

Unused keys for user shortcuts

```
?   Unused
[   Unused
]   Unused
{   Unused
}   Unused
Alt combinations are reserved for user's own shortcuts
```

Productivity Tips

Network rendering rules of thumb are:

- If some servers are slow, the box count should be 4-10 times the number of threads.
- Box division should be symmetric. 7*4 instead of 14*2 - it renders quicker as the boxes are "squarer".
- Put the slowest systems last in the server list defined by the render settings 'Boxes' tab. The fast systems have time to render the first boxes while the client is activating the slow systems.
- Save texture maps as JPG images for faster network render times.

Use analytical objects as much as possible in complex scenes - SDS and NURBS objects use larger amounts of memory.

Use the SHIFT key to constrain objects and tools to horizontal or vertical directions. The SHIFT key can also be used to size object proportionally when using object handles, and set rotation angles to 45 degree increments.

Snap to points and corners of objects with the '5' key.

On color swatches, right click to bring up the "expand" option. This allows you to quickly select the appropriate color using the color wheel.

Make use of clip maps to reduce scene complexity.

Use level structures to organize your work - drop multiple objects to their own level by multi-selecting with the SHIFT key, then right click on the select window and select "Drop to a level" from the popup menu. Projects organized with clever level use are far more manageable.

To change parameters of any sub objects - (e.g. change all objects in a level to "not selectable") simply select the level object, change the "not selectable" tag and all sub object are then not selectable. Likewise, to change all the objects in a scene to have an alpha value of 1 (to render to an alpha channel): Select the ROOT level, and change the 'Col/Surface properties/Alpha' value to 1 - all objects will be rendered solid against a clear background.

When panning and zooming objects, occasionally they may "disappear" in the view window. This is most likely due to the clipping planes utilized to speed up view refreshes. To see all the objects in a scene, simply drag the "Root" level icon into the view window. This sets the clipping planes to the extents of the scene and allows the user to see all objects. Conversely, dragging the object in question to the view zooms to the extents of that object/level.

Snap to edges and lines with the 6 key.

You can use CAD packages to model a complex geometry and use the IGES plugin to import it. This allows you to make use of CAD programs' inherent speed and accurate modeling tools, then import them into Realsoft 3D as splines, which can then be extruded or lofted into 3D shapes (Requires IGES plugin).

Use multiple Input Planes to align new objects with existing objects. Simply drag the IPlane onto the view to set it as the current IPlane - and make sure that under the view properties window "Lock to Input Plane" is set. To align the input plane to the current view angle, use the "Set Input Plane Orthogonal to View" button. To match the rotation of an object, simply copy the rotation angle from the object in question and paste it into the rotation angle field of the IP plane.

To set camera keyframes very quickly:

1. Create a camera from the view - select the camera button - then accept it straight away.

2. Under view properties dialog box, turn on 'Camera/tracking/track always'.

3. Turn on RECORD in the animation slider bar. (This is important - use "track always" with caution when NOT in animation recording as it saves the view position to the camera at all times.)

4. Move the time slider along halfway. Move the camera with the Alt/Right mouse button.

5. Move the time slider to the end. Move the camera with the Alt/Right mouse button.

6. Turn off RECORD. This has set 3 keyframe positions for the camera. Make sure you TURN OFF "track always" now before moving the view again.

7. Move the time slider and the view pans through the 3 keyframes you just set.



Note

Keep the movements relatively small between keyframes (<45 degrees per key) to keep the results predictable. The first keyframe is created automatically at frame 0.

One can save "selection sets" of meshes as separate objects by using the "group" tool. Edit the mesh - select the points - select "group" - accept.



Note

One can have as many sub-groups as required. Separate groups can have the same points inside.

To get soft shadows, create a point light and turn up the "size" field. Use the "Quality" setting to find a good balance of speed vs. output quality.

Boolean objects can be easily edited and animated. Simply turn off "wire invisible" of sub objects then move/size/rotate them.

To "depth" (3D) snap points - use the comma (,) key in conjunction with dragging.

To draw on an object - use "surface input" under view properties/input. This allows the user to draw curves on surfaces.

You can use the unified creation tools to create an SDS object that can then be edited. For example, SDS cubes and SDS rectangles are good starting points.

Use can convert NURBS meshes to SDS objects using the To SDS tool in the 'Construction' tab of the toolbar. Polygonal NURBS meshes are best for converting to SDS objects. Keep SDS objects as simple as possible at all times.

Troubleshooting

The ground plane is not visible

- Make sure that perspective projection is on ('Camera/Perspective' in view popup menu).
- Check that 'View properties/Input/Orientation guides/Ground plane' is set.
- Turn 'Grid draw' option off (disables ground plane drawing to avoid visual confusion between the two).
- Select 'View Orientation/Reset' from the view popup menu.

Cannot RMB-Ctrl-scroll a toolbar

- Move the mouse pointer over a toolbar area that does not activate a button, and check that 'Scrolled' is set using the toolbar's popup menu.

A window is missing

- Check that the window in question is set in the 'View' pulldown menu.
- Make sure that the window is not in a minimized state.
- Check that the window is included in the 'Windows' folder of the program installation (in case you have accidentally moved/deleted the file - if so, you may have to re-install suitable parts of the program).
- The window position may be outside the visible screen after loading a project from a higher resolution or a dual monitor system. To recover, add the startup parameter '-forcetoscreen' to the program's desktop icon or set the respective option using the Startopts program.

The screen is not updated correctly in OpenGL when moving/closing windows

- Update the driver library of your OpenGL accelerator.
- Switch to GDI mode and then back to OpenGL.

The program occasionally crashes in OpenGL mode

- Update the driver library of your OpenGL accelerator.
- Use the native wireframe drawing instead of OpenGL.

Geometric objects

Cannot see an object or parts of the object are not visible

- Open the property window and make sure that 'Gen/Invisible in real time rendering' and 'Wire/Hide Geometry' are not set.
- Select the object and use 'Camera/Auto Focus' from the view's popup menu. The object may be too far or behind the camera (outside the clip range).

Part of a fractal plant is missing and an error message 'maximal plant complexity reached' appears

- Open the property window, select the plant and increase the Maximal complexity value.

Fractal plant's UV coordinates are not continuous (form a checkered pattern)

- Open the property window, select the plant and enable the Set UV option.

Boolean operations

Components of a Boolean operation are not visible

- Make sure that none of the Boolean sub objects has 'Property Window/Gen/Hollow' set.
- Verify that use of 'Volume Inverted' object property is consistent. This option may become unintentionally set when experimenting with Booleans and moving objects around in hierarchy.

A surface renders with a noisy random pattern

- You have performed a Boolean operation where two cutting surfaces overlap exactly. Move either of the components slightly (1 micrometer is enough), make sure that the 'drill' is long enough, etc.

Rendering

An object is missing from ray traced images

- Select the object and check that 'Property Window/Gen/Invisible in real time rendering' is not set (check also the parent levels).

A parallel mapped texture renders with a noisy random pattern on the target surface

- If you are using 'Finite' options of the parallel mapping, the target surface should be properly inside the mapping cube.

File rendering shows a different image than the view window

- Make sure that the project includes a camera object, and that the camera and view orientations match. Set 'Current' option for the desired camera object, in case there are multiple cameras included.

The file rendered image appears stretched

- Verify that the aspect ratio of the current camera object and the output resolution are consistent (taking into account the pixel aspect ratio of the planned target media).

Cannot see through transparent objects, or parts of transparent objects appear black

- Make sure that 'Recursion Depth' of the rendering settings you are using is high enough. For example, 'Speed over quality' does not show transparency.

Cannot change the color of a backdrop object

- A backdrop surface is usually 'self-illuminating' - light sources do not affect it. To change the color, open the property window and go to the 'Col' tab. Select 'Illumination' from the 'Attribute' property list and define a new illumination color using the gadget below the property list.

NURBS curves are not visible in rendering

- Check that 'Property Window/Gen/Invisible in photo realistic rendering' is not set.

Rendering halts immediately and gives an error message 'channel xxx not found'

- Go to the Channels tab of the select window, create a new channel and rename it as specified in the error message.

Network rendering

Cannot use network rendering

- Windows95/98 have limitations in TCP/IP services required by network rendering.

Network rendering is slow

- Save texture maps as JPG images for fastest network render times.
- Check network configurations - the client may spend time in unsuccessfully trying to connect to a server.

A server system in network rendering cannot render the current project

- Check the texture file paths of the project. For example, a hard disk may be mapped to another letter in the server system as in the client.

Materials

A transparent material renders white

- Add 'Color=Constant(0,0,0) to surface properties. The default diffuse color is always white, and you have to eliminate diffuse shading.

Cannot see effect of a fog-like material

- Fog effects are heavily scale dependent. A fog material that looks good when mapped to a one meter wide object is practically invisible when mapped to a candle flame. Adjust the 'Turbidity' or 'Density' parameter of the fog.
- Check the rendering settings. Volumetric options should be set and recursion depth should be high enough (try a value greater than 2).

Animation

The camera does not keep its focus in the desired point after key framed rotations

- Set the pivot point of the camera to the desired center of rotation before key framing the camera.

A key frame-animated object rotates to the opposite direction to what was intended

- Add a key frame in the middle. Rotation chooses the shortest way to new positions and the new key shows the right way.

Simulation does not affect the objects

- Check that the parent level has 'Properties/Spec/Simulation' enabled.

An object does not follow exactly a given path

- Path animation keeps the initial offset between the object and the path. Select the object and the path, then apply the Unmap tool of the 'Lattice Mapping' tool tab. Then move the object to the correct position (use point snapping if necessary). Re-map the object to the path.



Chapter 15. Copyright

Real 3D is a trademark of Realsoft Oy, Finland. **Realsoft 3D** is a trademark of Realsoft Graphics Oy, Finland. 1989 ... 2009, All rights Reserved.

Realsoft 3D supports GIF format through GD library.

GD COPYRIGHT STATEMENT FOLLOWS THIS LINE

Portions copyright 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004 by Cold Spring Harbor Laboratory. Funded under Grant P41-RR02188 by the National Institutes of Health. Portions copyright 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004 by Boutell.Com, Inc.

Portions relating to GD2 format copyright 1999, 2000, 2001, 2002, 2003, 2004 Philip Warner.

Portions relating to PNG copyright 1999, 2000, 2001, 2002, 2003, 2004 Greg Roelofs.

Portions relating to gdttf.c copyright 1999, 2000, 2001, 2002, 2003, 2004 John Ellson (ellson@graphviz.org).

Portions relating to gdft.c copyright 2001, 2002, 2003, 2004 John Ellson (ellson@graphviz.org).

Portions relating to JPEG and to color quantization copyright 2000, 2001, 2002, 2003, 2004, Doug Becker and copyright (C) 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004 Thomas G. Lane. This software is based in part on the work of the Independent JPEG Group. See the file README-JPEG.TXT for more information.

Portions relating to GIF compression copyright 1989 by Jef Poskanzer and David Rowley, with modifications for thread safety by Thomas Boutell.

Portions relating to GIF decompression copyright 1990, 1991, 1993 by David Koblas, with modifications for thread safety by Thomas Boutell.

Portions relating to WBMP copyright 2000, 2001, 2002, 2003, 2004 Maurice Szmurlo and Johan Van den Brande.

Portions relating to GIF animations copyright 2004 Jaakko Hyvätti (jaakko.hyvatti@iki.fi)

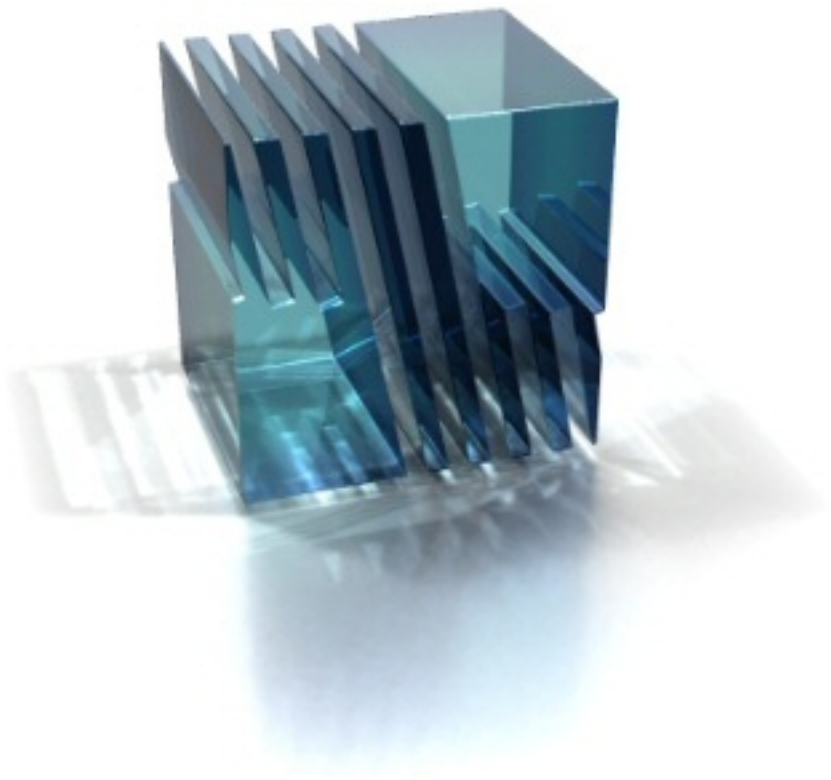
Permission has been granted to copy, distribute and modify gd in any context without fee, including a commercial application, provided that this notice is present in user-accessible supporting documentation.

This does not affect your ownership of the derived work itself, and the intent is to assure proper credit for the authors of gd, not to interfere with your productive use of gd. If you have questions, ask. "Derived works" includes all programs that utilize the library. Credit must be given in user-accessible documentation.

This software is provided "AS IS." The copyright holders disclaim all warranties, either express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to this code and accompanying documentation.

Although their code does not appear in the current release, the authors also wish to thank Hutchison Avenue Software Corporation for their prior contributions.

END OF GD COPYRIGHT STATEMENT



Chapter 16. Imprint

Realsoft
Polvikatu 3 C 22, 33100 Tampere, Finland
http://www.realsoft.com

Realsoft 3D User Manual Sixth Edition, December 2009. Copyright 2009 Realsoft. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means without permission in writing from Realsoft.

All images are copyrighted by their respective owners. All rights reserved.

Front Knife image Copyright (BT)-grafik
Learning the Basics Copyright (BT)-grafik
Modeling Copyright (BT)-grafik
Subdivision Surfaces Copyright Gunnar Radeloff
NURBS Copyright Gunter Hagedorn
Modeling Buildings Copyright Pixel Perfect
Materials Copyright (BT)-grafik
Rendering Copyright Gunnar Radeloff
Post Effects Copyright Gunnar Radeloff
Animations Copyright Gunnar Radeloff
Simulations Copyright (BT)-grafik
User Interface Copyright (BT)-grafik
Scripting Languages Copyright Martin Weber
Miscellaneous Topics Copyright (BT)-grafik
Design Copyright (BT)-grafik

Index

Symbols

3D Painting, 225
3D Snapping, 80

A

Acceleration, 391
Accuracy, 82
Airplane, 414
Alpha Channel, 305
Aluminium, 250
Ambient Light, 292
Anchoring, 386
Angle Constraints, 383
Angle Grid:, 74
Animate UV, 378
Animated Material, 371
Animations,
Atmosphere, 296
Available GUI Objects, 418
Available GUI Objects Window, 418
Available Objects Window, 418
Average of Materials, 255

B

Background Color, 108
Banana, 165, 173
Batch Rendering, 393
Bezier, 164
Bi-Rail, 176
Bird, 134
Blending Materials, 218
Blur Effect, 335
Boolean Operations, 83
Boundary Surface Solids, 87
Breaking Curves, 160
Breaking Meshes, 168
Bubbles, 400
Bump Mapping, 239

C

Cabinet Group, 212
Camera, 106
Camera Backdrop, 108
Camera Lens Flare, 108
Camera Tracking, 109
Cauge, 115
Caustics, 317
Centripetal, 163
Chord Length, 163
Choreography Parametrization, 388
Clip Mapping, 262
Clouds, 296
Coffee Cup, 124
Collect External Files, 77
Collision Deformations, 406
Collision Detection, 406
Color Animation, 352
Color control, 277
Compass Macros, 427
Compass Simulation, 410
Compositing, 338
Concatenate, 163
Constraint Script, 114
Creases, 148
Creator, 398
Cross Sections, 177
Current Project, 76
Current Working Project, 76
Curve Order, 158
Curve Weight, 161
Curves, 157
Curves from Surfaces, 191

D

Dart, 412
Decorating, 30
Degree Elevation, 170
Depth of Field, 109
Dirt, 273
Displace Mesh, 192
Distributed Rendering, 286
Doors, 28
Drag Box, 78
Drag Box Selector, 78
Dragging, 78

E

Editing Key Frames, 353
Elasticity, 407
Euler Angles, 377
Event Driven Animation, 388
Explosions, 401
Extrude Curves, 179

F

Face Materials, 152
Fade Mapping, 263
Falloff, 291
Field Evaluator, 115
Field Rendering, 301, 380
File Rendering, 283
Fillets, 187
Fire, 298
Flashlight, 289
Floors, 209
Fluid, 120
Fluid Dynamics, 411
Fog Examples, 242
Font Curves, 110
Font Tool, 188
Foot Steps, 387

Frame Count, 354
Frame Range, 356
Freehand Curves, 158
Freehand selector, 80
Friction, 408
Fur, 245
Furniture, 32

G

Galaxy, 405
GI in Animations, 320
Glass, 248
Global Fog, 376
Global Illumination, 312
Gravity, 403
Grids, 82
Ground Floor, 209
GUI Customization, 418
GUI of VSL Material, 238

H

Hair, 50
Handles, 71
Harry, 35
Hierarchical Animations, 364
Hipped roof, 206
Hot Keys, 457

I

Illumination Map, 307
Image Vectorization, 190
Indirect Illumination, 310
Inertia, 407
Instances, 113

J

JavaScript, 432

K

Key Bindings, 420
Key Driven Animation, 388
Key Frame Animation, 351
Keyboard Shortcuts, 457
Kitchen, 212

L

Lake, 266
Landscape, 34
Lasso, 80
Lasso Selector, 80
Lattice Mapping, 89
Layers, 451
Leaves, 103
Lens Flare Animation, 343
Level Material, 224
Light Falloff Curve, 246
Lighting, 289
Link Object, 121
Localization, 428

M

Macro Button, 425
Macro Keys, 420
Macro Menus, 425
Macros, 94
Magnetism, 410
Magnifying glass, 318
Map2Obj, 154
Mapped Shadows, 308
Material drag and drop, 214
Material Mapping, 214
Materials,
Matrices, 446
Matte Shadows, 245
Measuring Units, 82
Metaball, 117
Motion Blur, 299, 380
Move Handle, 71
Multiple Cameras, 107
Multiple Floors, 209
Muscles, 390

N

New Classes, 452
Numeric Modeling, 81
NURBS Mesh, 166
NURBS Parametrization, 163

O

Orbiting Planets, 403
Outline Rendering, 314

P

Paper, 244
Parametric Objects, 194
Particle Effects, 333
Particles, 97
Path Animation, 366
Pendulum, 404
Periodic Animation, 362
Pivot, 73
Pivot Point, 73
Plasma, 242
Point Snapping, 78
Post Shading, 345
Pot, 256
Previewing Materials, 265
Project, 76
Propeller, 147

Q

Quaternions, 376

R

Rational SDS, 145
Rebound, 409

Render Box, 285
Rendering Settings, 283
Rigging, 384
Rigidity, 407
Roof, 31
Roof Line, 206
Roof Modeling, 206
Rotate Handle, 72
Rotated Surfaces, 172
Rotation Handle, 72
Rotation Handles, 72
Rotations, 376
Rotator, 111

S

Saving a project, 77
Scale Handle, 72
Scale Handles, 72
Scaling Handle, 72
Scaling Time Lines, 353
Scanline, 302
Scope Mapping, 263
Scripting,
Scripting Languages, 431
SDS Compass, 128
SDS Handles, 130
SDS Knife, 155
SDS Rounding, 149
Shadows in Fog, 243
Sharp Corner, 160
Shinbone, 364
Simulations,
Sink, 151
Skeletons, 381
Slide Projector, 246
Smoke, 336
Snapping to Curve, 81
Soft Shadows, 290
Special Light, 293
Spin, 397
Steel, 248
Subdivision Surfaces,
Swap, 170
Sweep, 175
Swimming Fish, 367
Symmetric Modeling, 141

T

Technical Drawing, 348
Texture Mapping, 221
Thighbone, 364
Time Code, 354
Timing, 368
Toolbar Editing, 417
Torus, 173
ToSDS Tool, 136
Transformation Handle, 71
Transformation Handles, 71
Translation Handle, 71
Translation Handles, 71
Trim Booleans, 182
Trimming, 180
Troubleshooting, 466
Tube Object, 112
Type Casting, 251

U

Unified Tools, 65
Uniform, 163
Uniform Scaling, 74
User Interface,
UV Editing, 153

V

V3 Material, 225
Vectors, 444
Velocity, 397
Vertex Variables, 140
Video System, 354
Visible Lights, 297
Volumetric Rendering, 294
VSL Examples, 233
VSL Intro, 233
VSL Material, 224
VSL Procedures, 249
VSL Wizards, 238

W

Wall, 276
Waterfall, 379
Weight of Choreography, 362
Welding, 184
Window, 194
Windows, 28